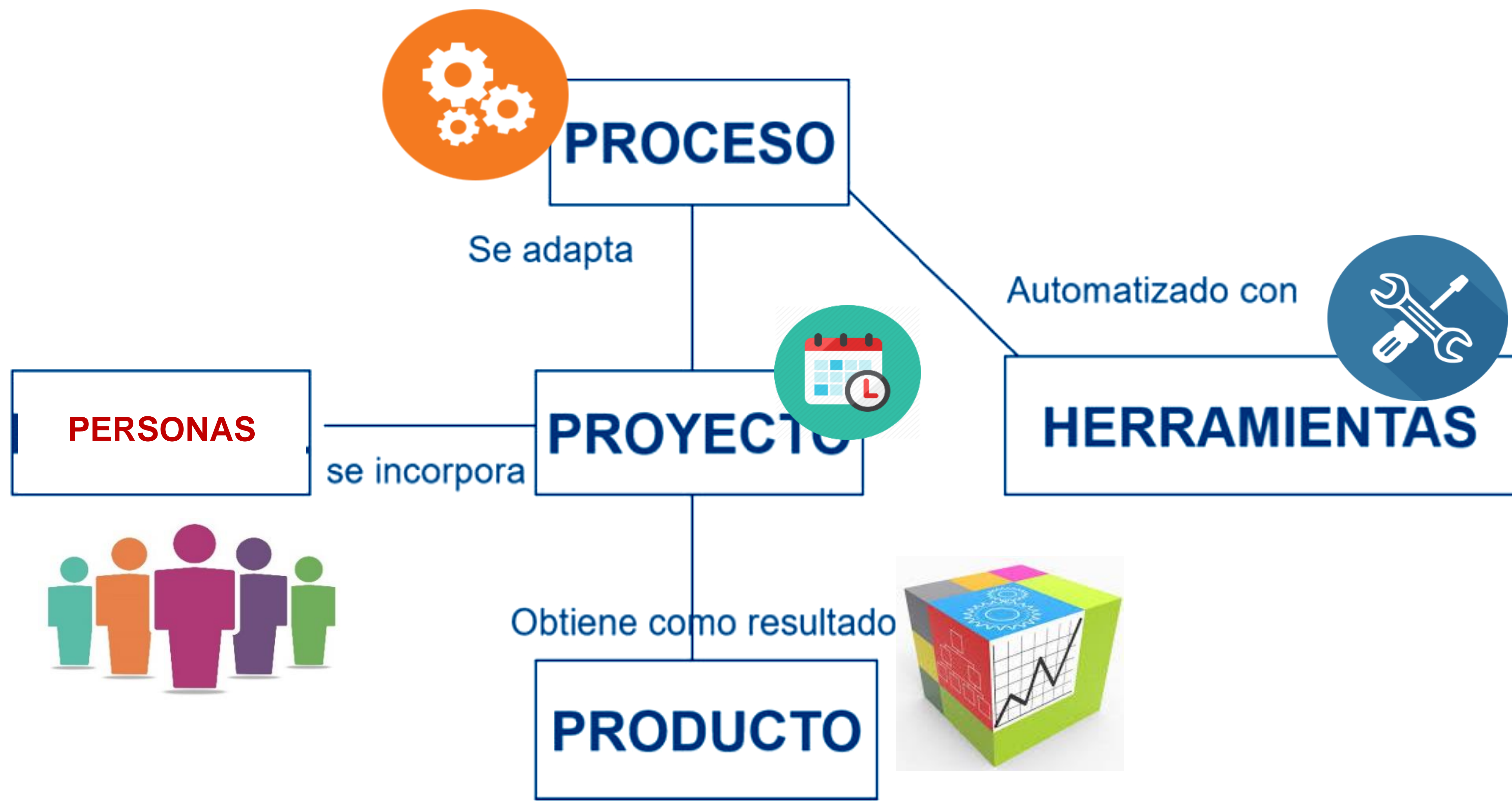


Universidad Tecnológica Nacional  
Facultad Regional Córdoba  
Cátedra de Ingeniería y Calidad de Software  
Docentes: Judith Meles- Laura Covaro

# Software Configuration Management (SCM) (o más allá del Commit, Update)



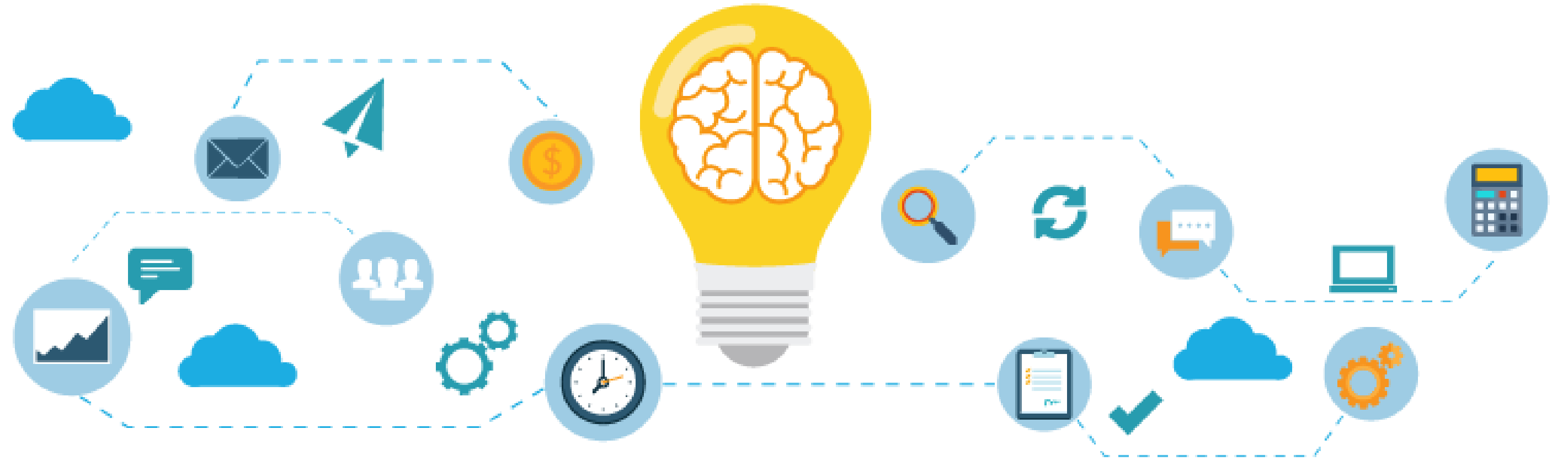
# Software en contexto



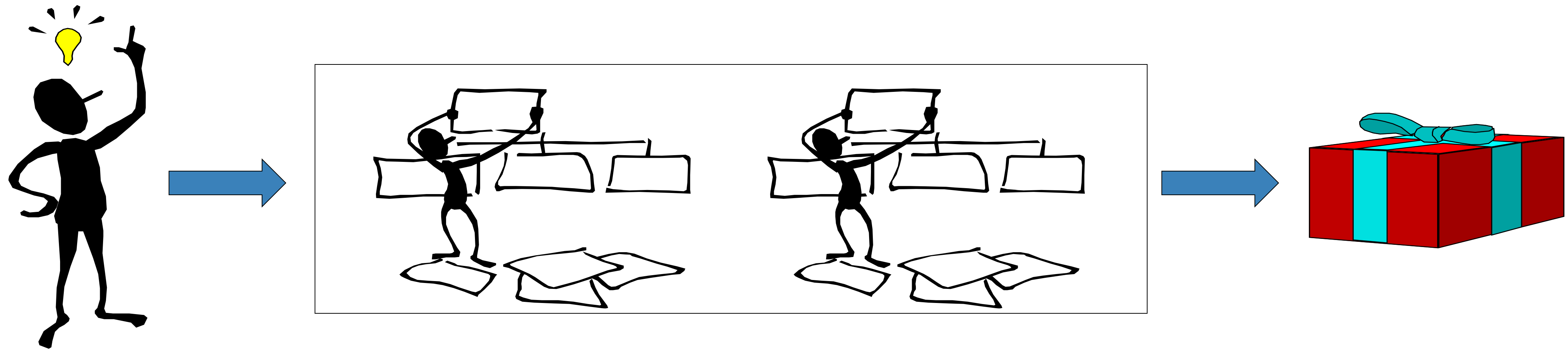
# ¿Cuándo pensamos en Software... en qué pensamos?

**Conjunto** de:

- *Programas*
- *Procedimientos*
- *Reglas*
- *Documentación*
- *Datos*



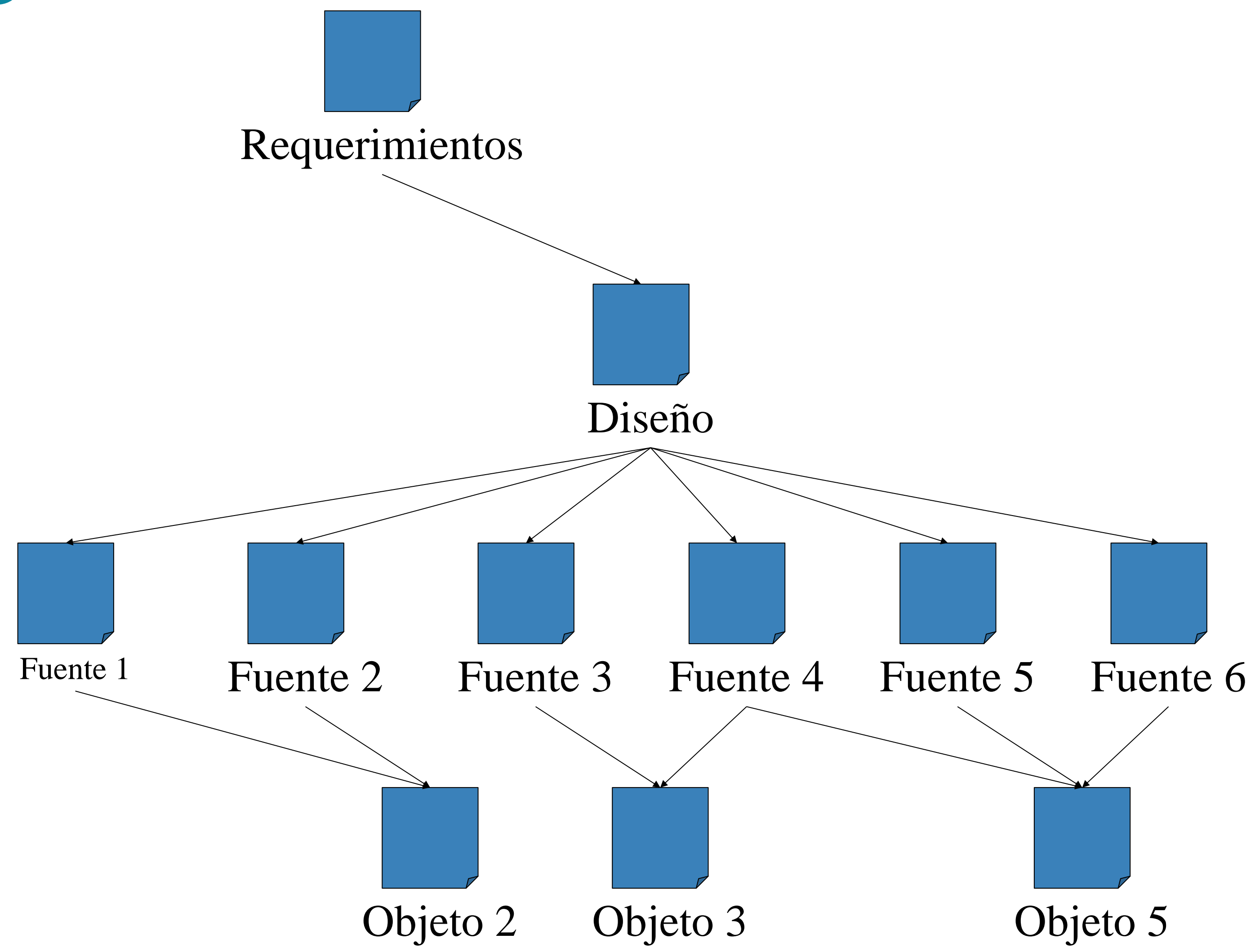
# El Software



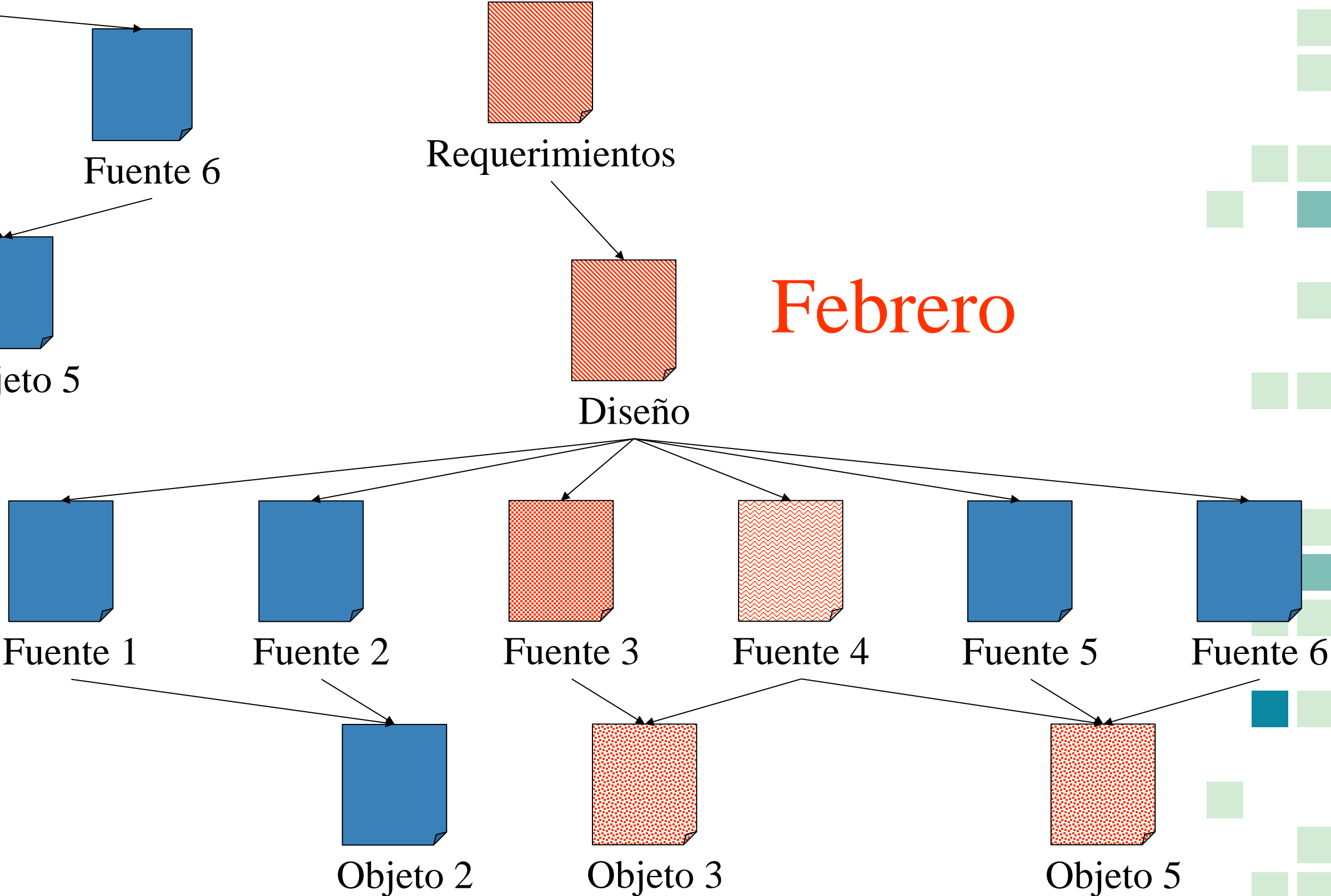
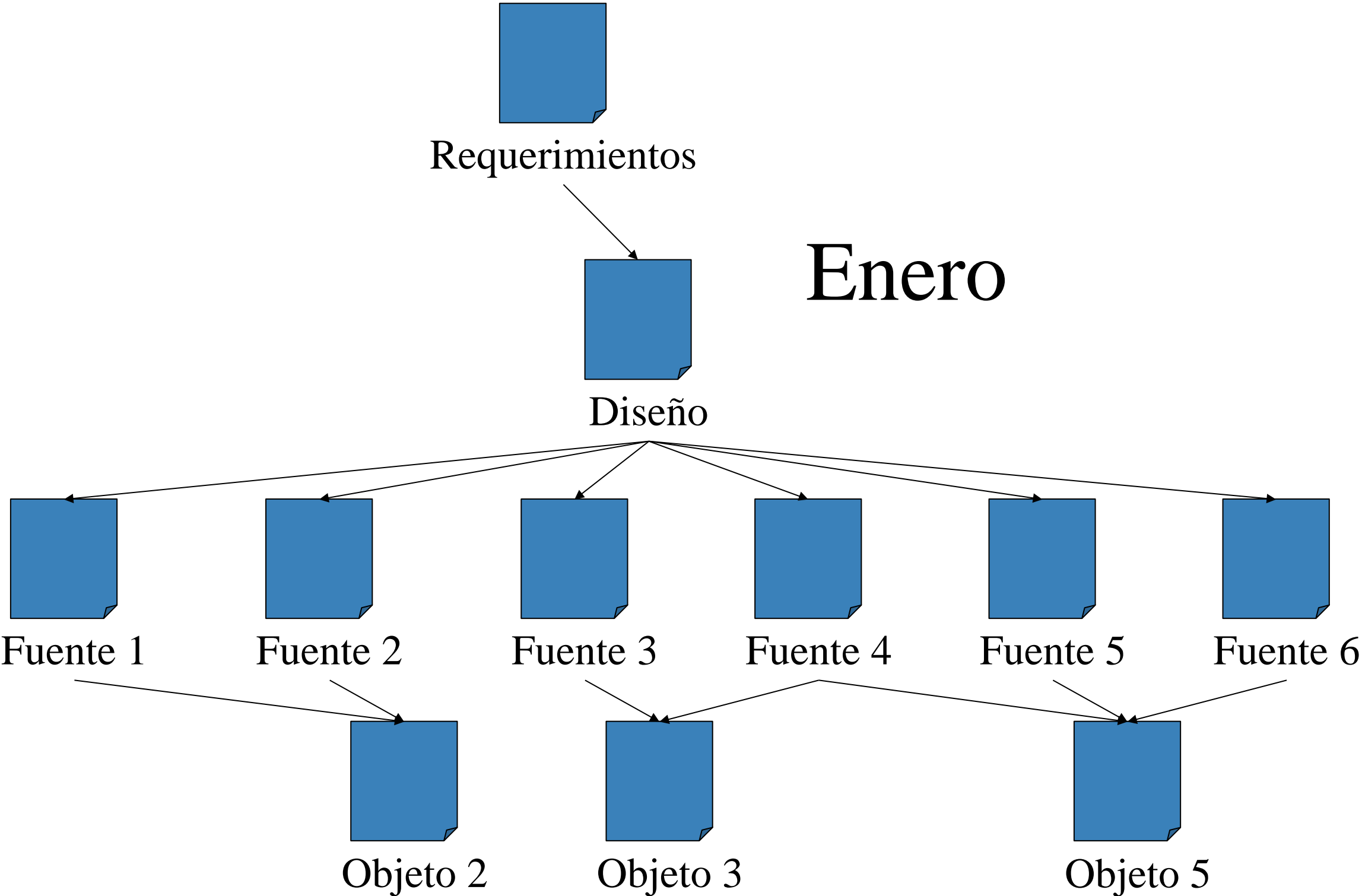
- Información:
  - estructurada con propiedades lógicas y funcionales.
  - creada y mantenida en varias formas y representaciones.
  - confeccionada para ser procesada por computadora en su estado más desarrollado

# El software

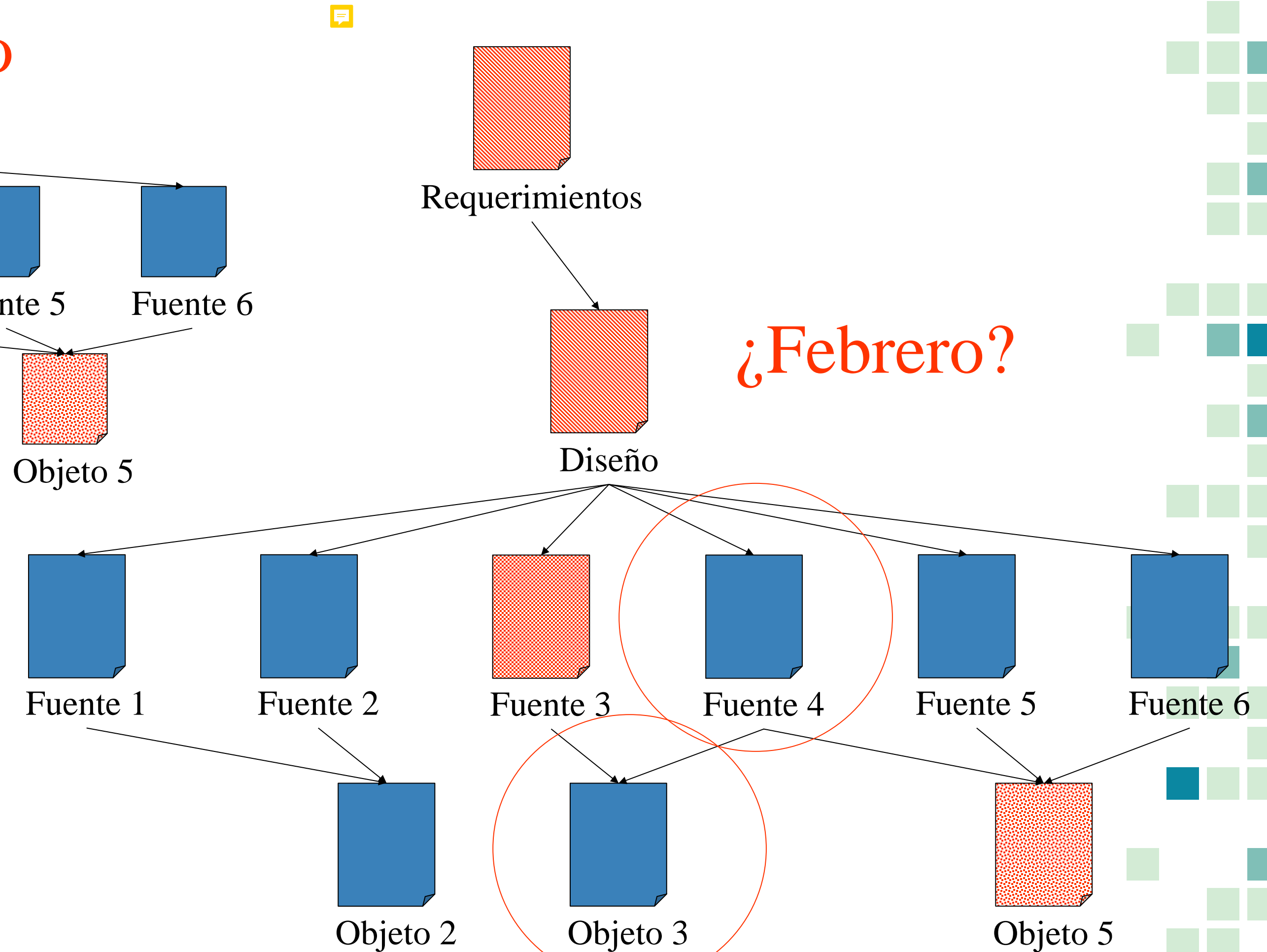
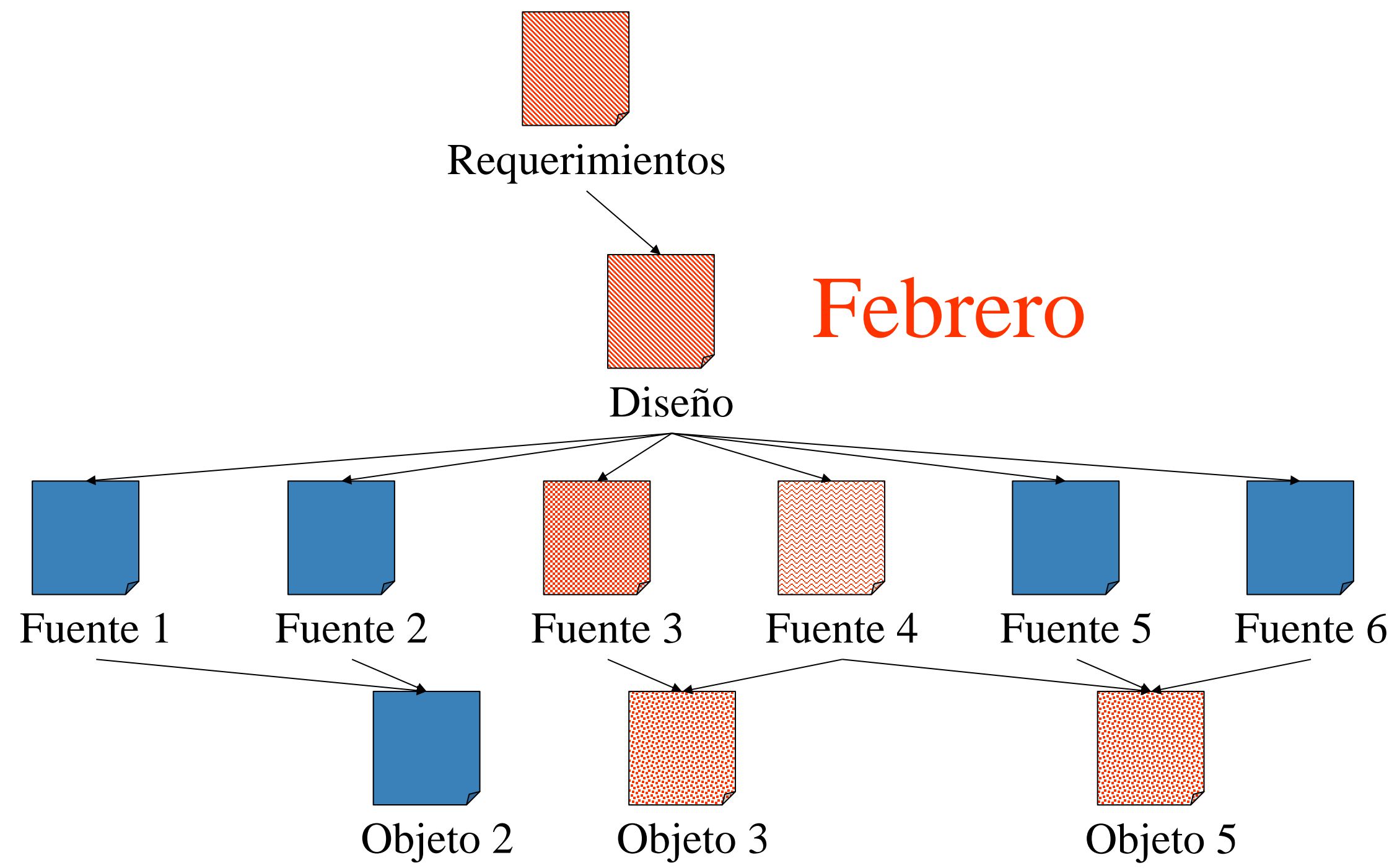
Enero



# La evolución del software



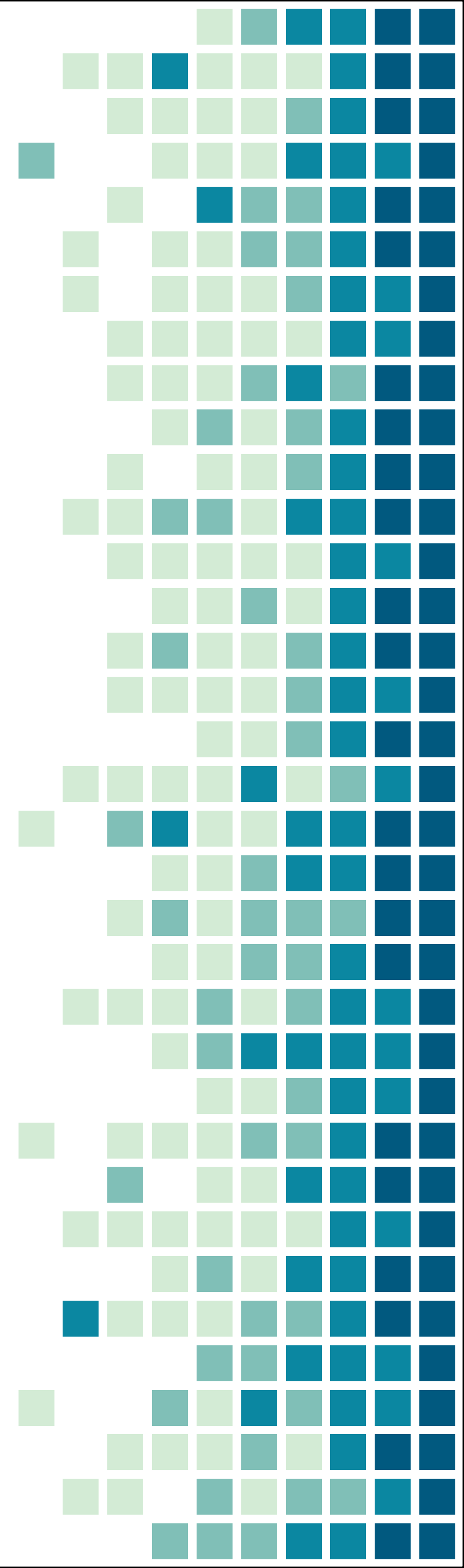
# Problemas



# Cambios en el Software

Tienen su origen en:

- ❖ Cambios del negocio y nuevos requerimientos
- ❖ Soporte de cambios de productos asociados
- ❖ Reorganización de las prioridades de la empresa por crecimiento
- ❖ Cambios en el presupuesto
- ❖ Defectos encontrados a corregir
- ❖ Oportunidades de mejora





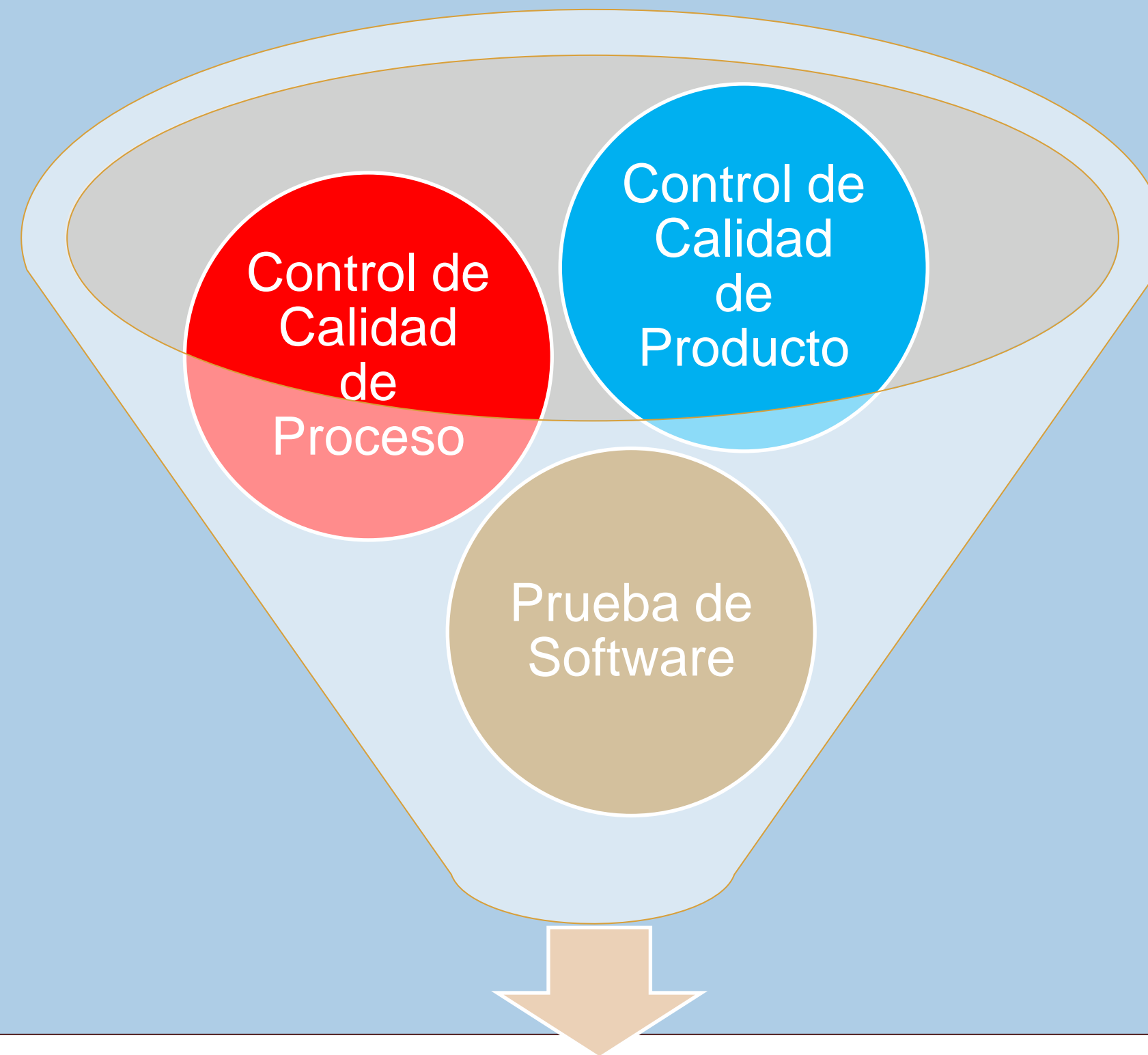
# SCM como disciplina de soporte

Es una actividad “paragüas”, transversal a todo el proyecto, relevante para el producto a lo largo de su ciclo de vida.

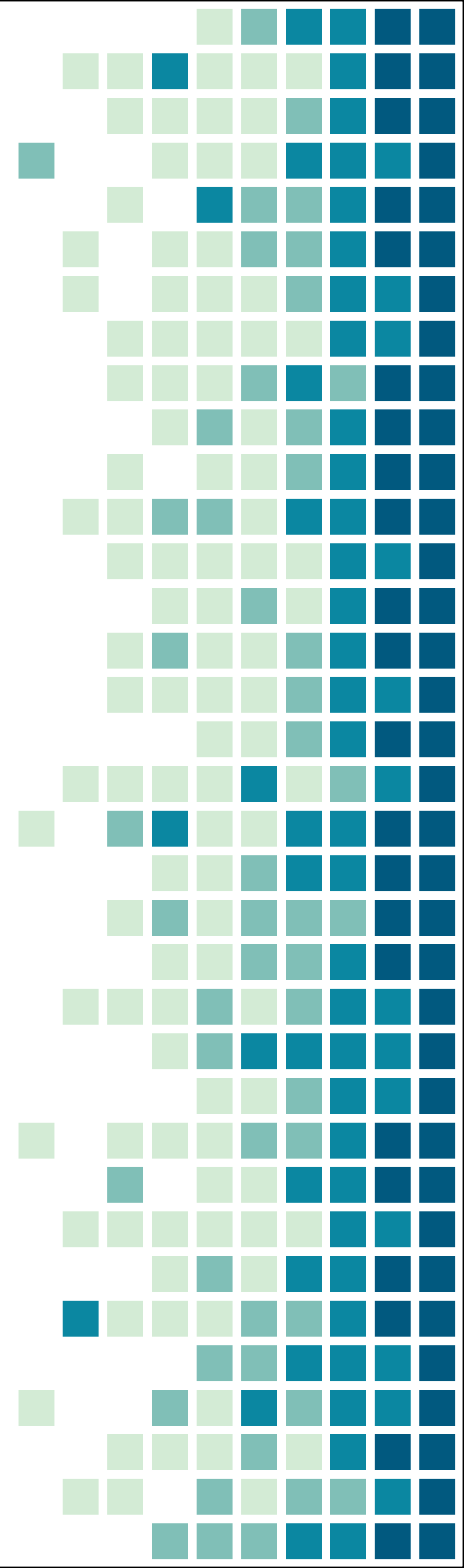


# Disciplinas de soporte del Software

Administración de  
Configuración de Software



**Aseguramiento de  
Calidad de Software**



# Un poco de Historia



Tiene su origen a mediados de 1950s, cuando CM (por Configuration Management) originalmente utilizado para desarrollo de hardware y control de producción, fue utilizado en el desarrollo de software.

# Definición

Una disciplina que aplica dirección y monitoreo administrativo y técnico a: identificar y documentar las características funcionales y técnicas de los ítems de configuración, controlar los cambios de esas características, registrar y reportar los cambios y su estado de implementación y verificar correspondencia con los requerimientos

(ANSI/IEEE 828, 1990)

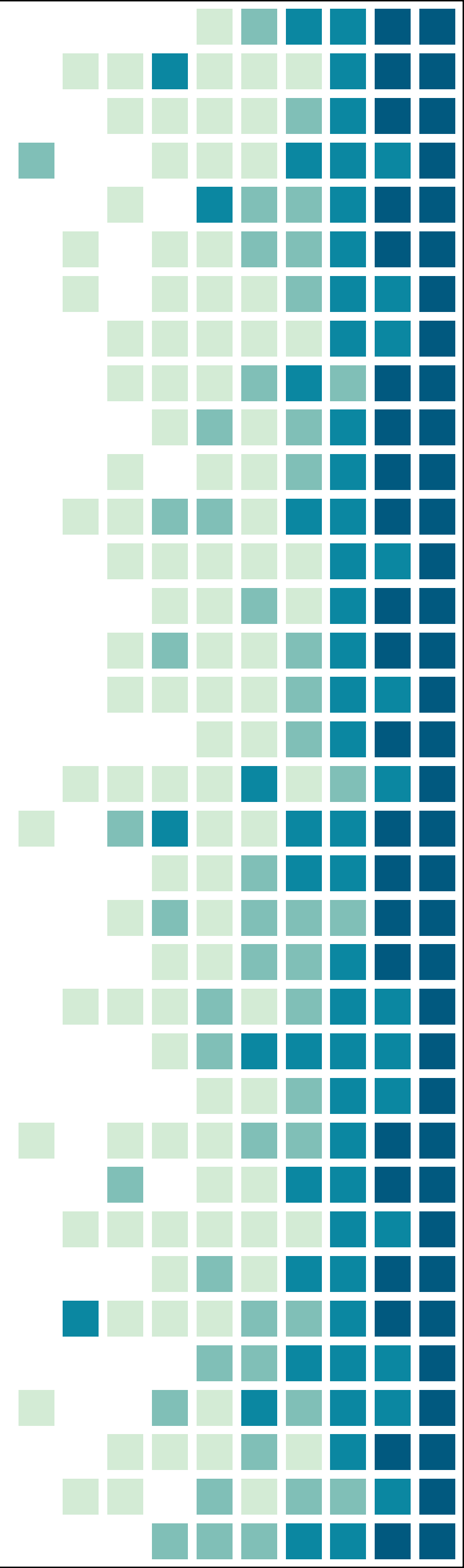


# ¿Por qué deberíamos gestionar la configuración del software?

Su propósito es establecer y mantener la integridad de los productos de software a lo largo de su ciclo de vida.

Involucra para la configuración:

- ❖ Identificarla en un momento dado
- ❖ Controlar sistemáticamente sus cambios
- ❖ Mantener su integridad y origen

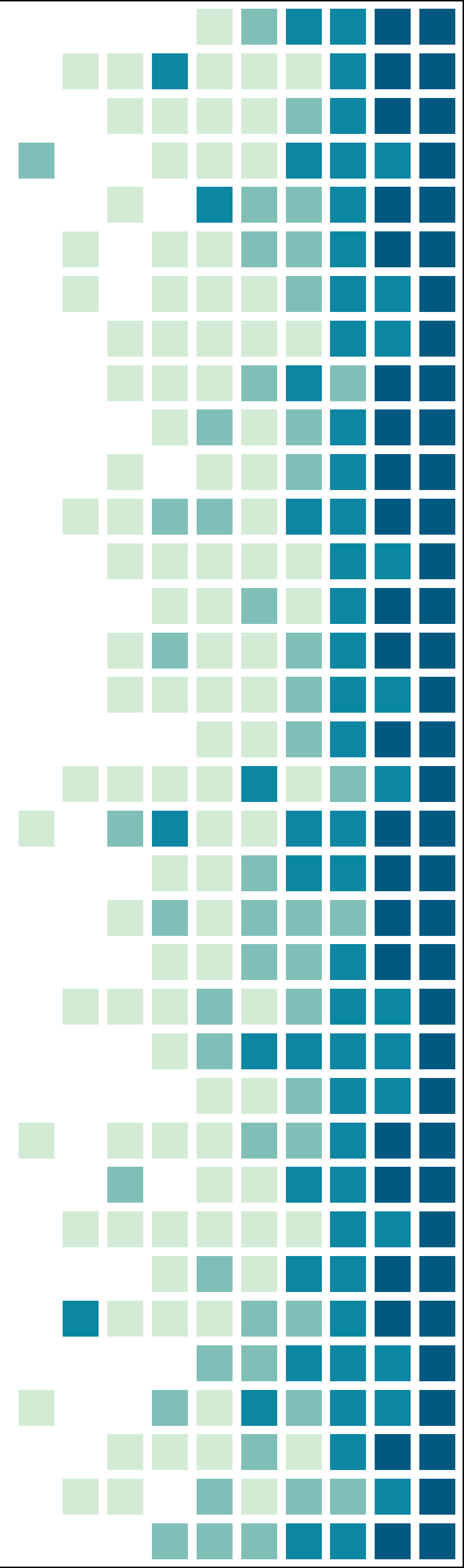
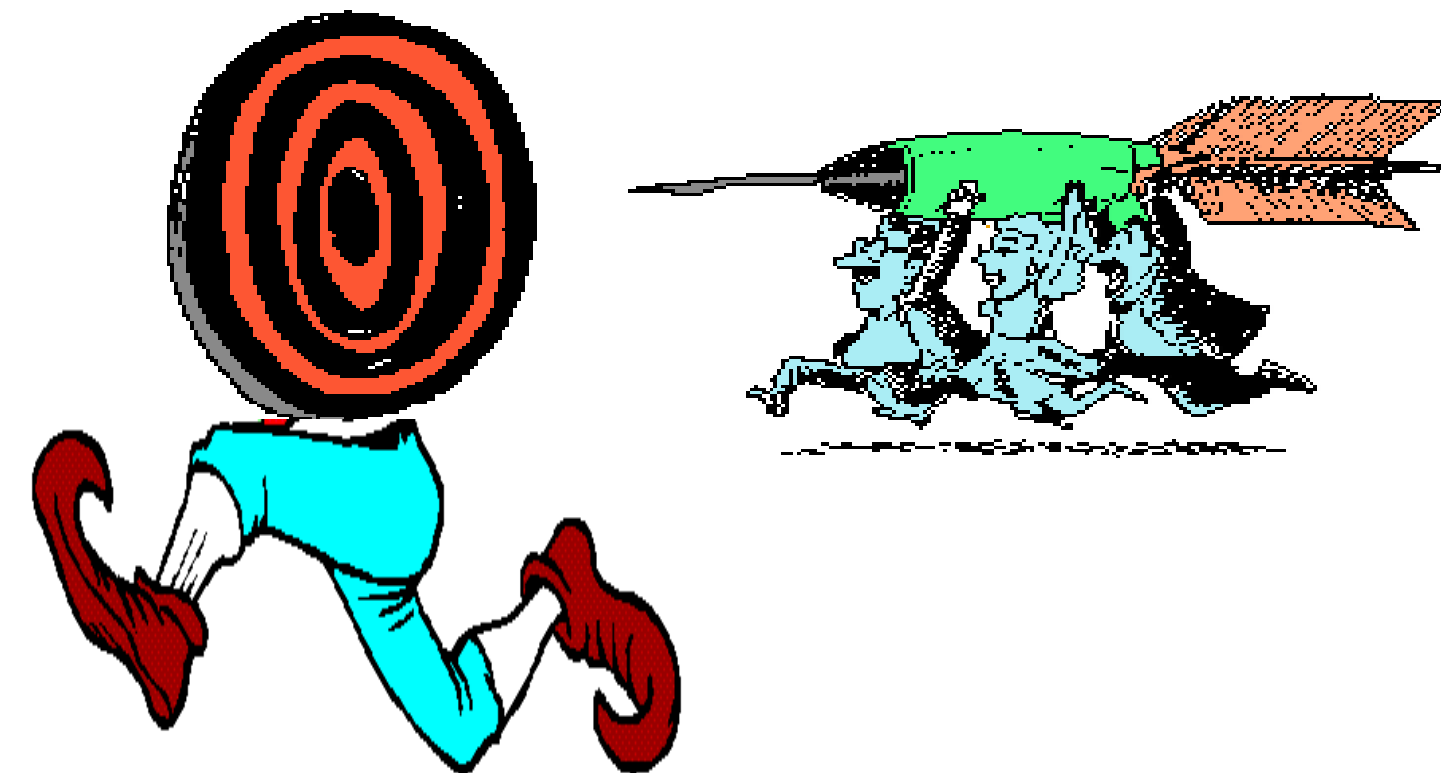




# Integridad del Producto

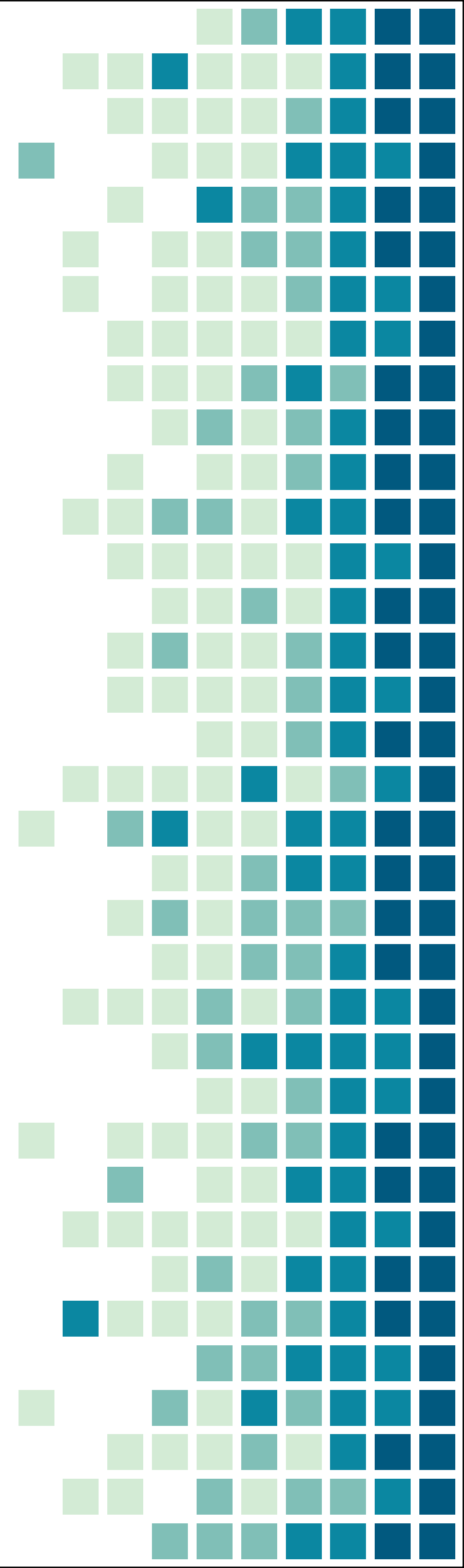
- satisface las necesidades del usuario
- puede ser fácil y completamente rastreado durante su ciclo de vida
- satisface criterios de performance
- cumple con sus expectativas de costo

El software: un blanco móvil



# Problemas en el manejo de componentes

- ❖ Pérdida de un componente
- ❖ Pérdida de cambios (el componente que tengo no es el último)
- ❖ Sincronía fuente - objeto – ejecutable
- ❖ Regresión de fallas
- ❖ Doble mantenimiento
- ❖ Superposición de cambios
- ❖ Cambios no validados



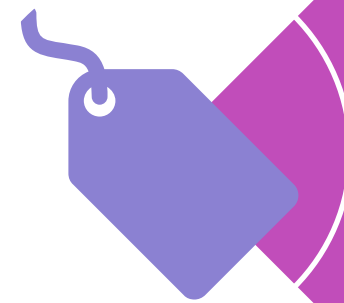
# Algunos Conceptos Clave para la Gestión de Configuración de Software



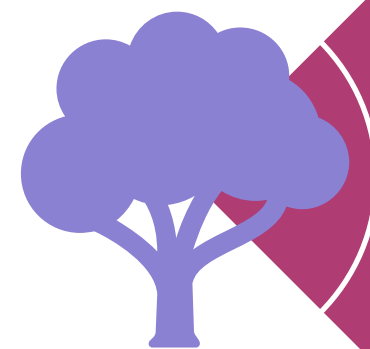
Ítem de Configuración



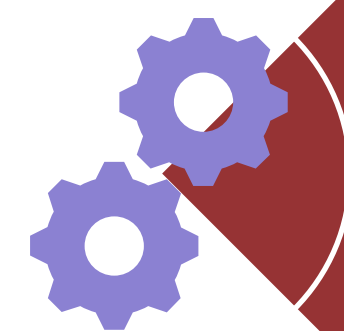
Repositorio



Línea Base



Ramas (Branch)



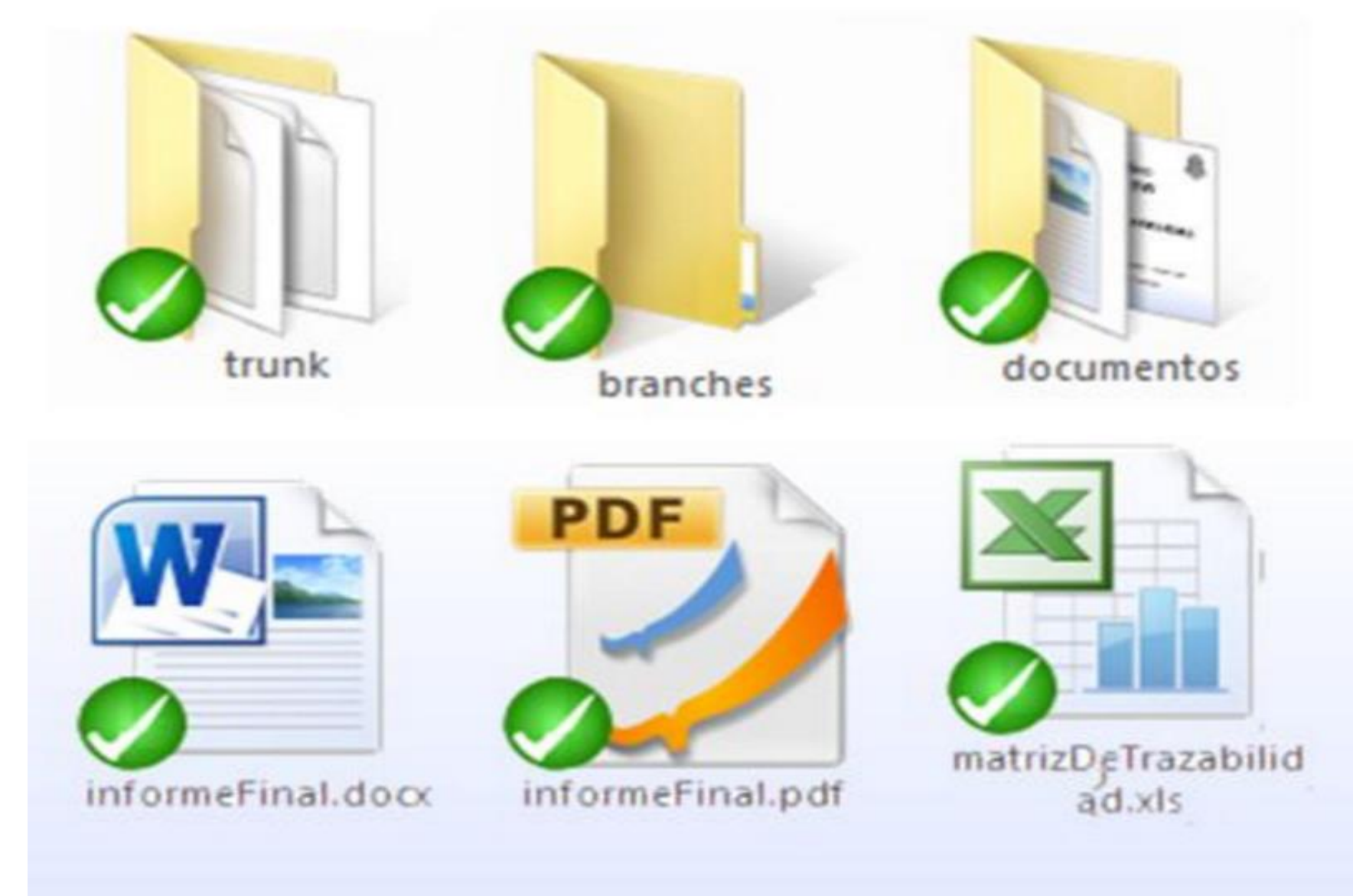
Configuración del Software



# Ítem de Configuración de Software (SCI)


- ✓ Documentos de diseño, código fuente, código ejecutable, etc.

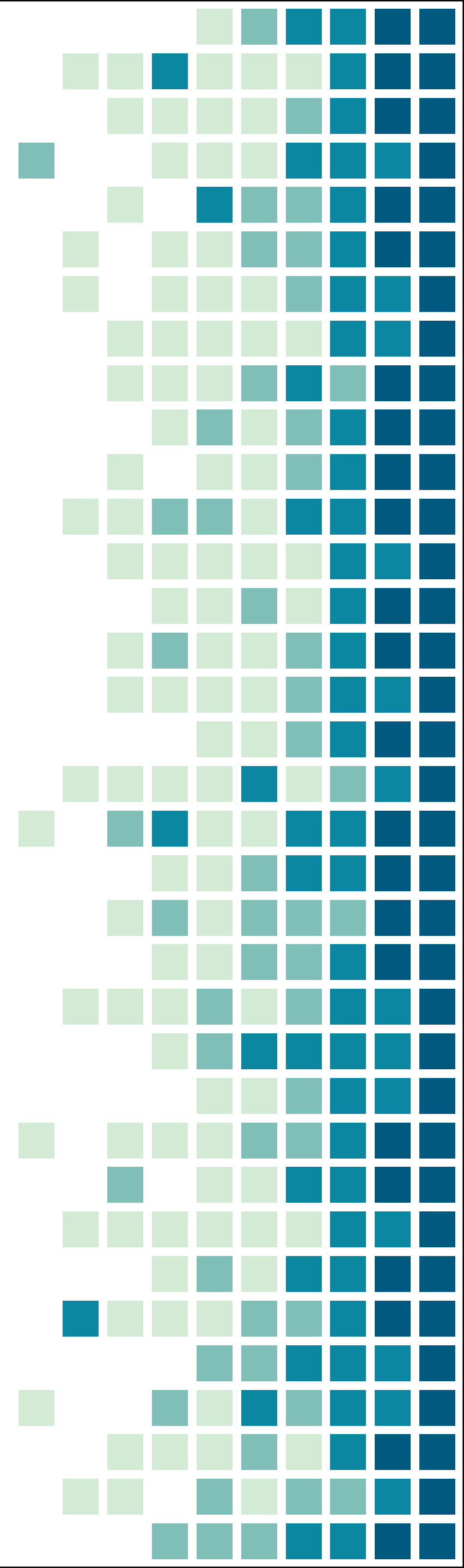
Se llama **ítem de configuración** (IC) a todos y cada uno de los **artefactos** que forman parte del producto o del proyecto, que pueden sufrir cambios o necesitan ser compartidos entre los miembros del equipo y sobre los cuales necesitamos conocer su estado y evolución.



# Algunos ejemplos de Ítems de Configuración

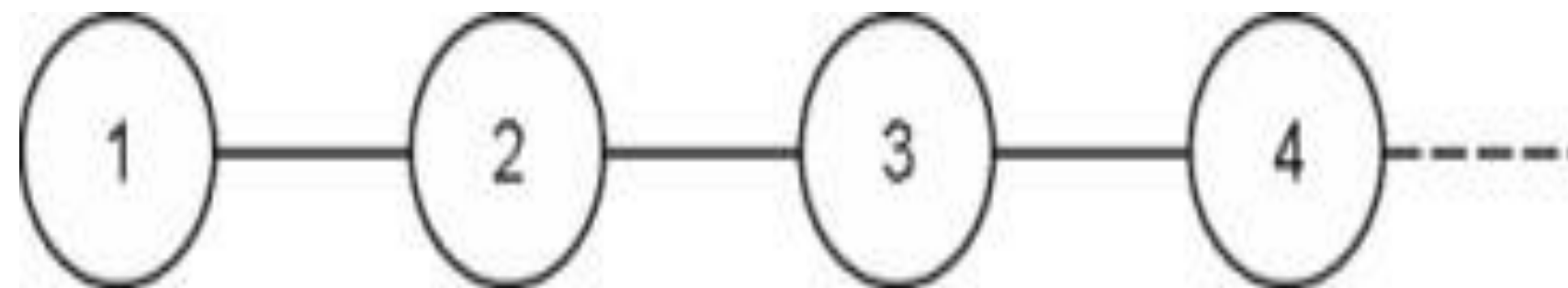


- ❖ Plan de CM
- ❖ Propuestas de Cambio
- ❖ Visión
- ❖ Riesgos
- ❖ Plan de desarrollo
- ❖ Prototipo de Interfaz
- ❖ Guía de Estilo de IHM
- ❖ Manual de Usuario
- ❖ Requerimientos 
- ❖ Plan de Calidad
- ❖ Arquitectura del Software
- ❖ Plan de Integración
- ❖ Planes de Iteración
- ❖ Estándares de codificación
- ❖ Casos de prueba
- ❖ Código fuente
- ❖ Gráficos, iconos, ...
- ❖ Instructivo de ensamble
- ❖ Programa de instalación
- ❖ Documento de despliegue
- ❖ Lista de Control de entrega
- ❖ Formulario de aceptación
- ❖ Registro del proyecto

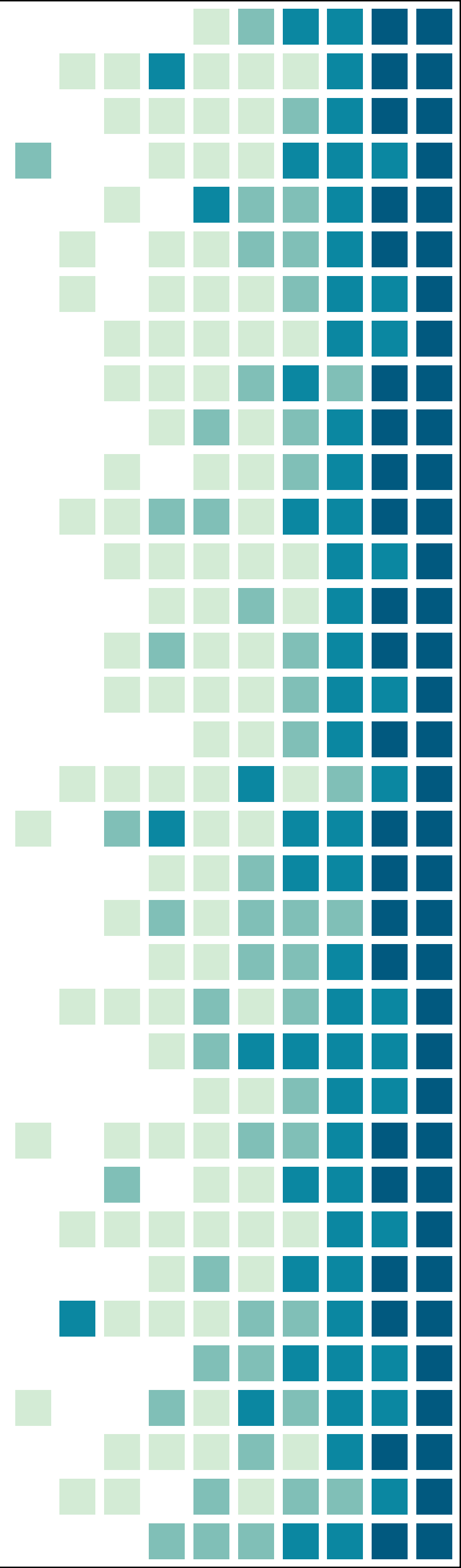


# Versión

- Una **versión** se define, desde el punto de vista de la evolución, como la forma particular de un artefacto en un instante o contexto dado.
- El control de versiones se refiere a la evolución de un único ítem de configuración (IC), o de cada IC por separado.
- La evolución puede representarse gráficamente en forma de grafo.

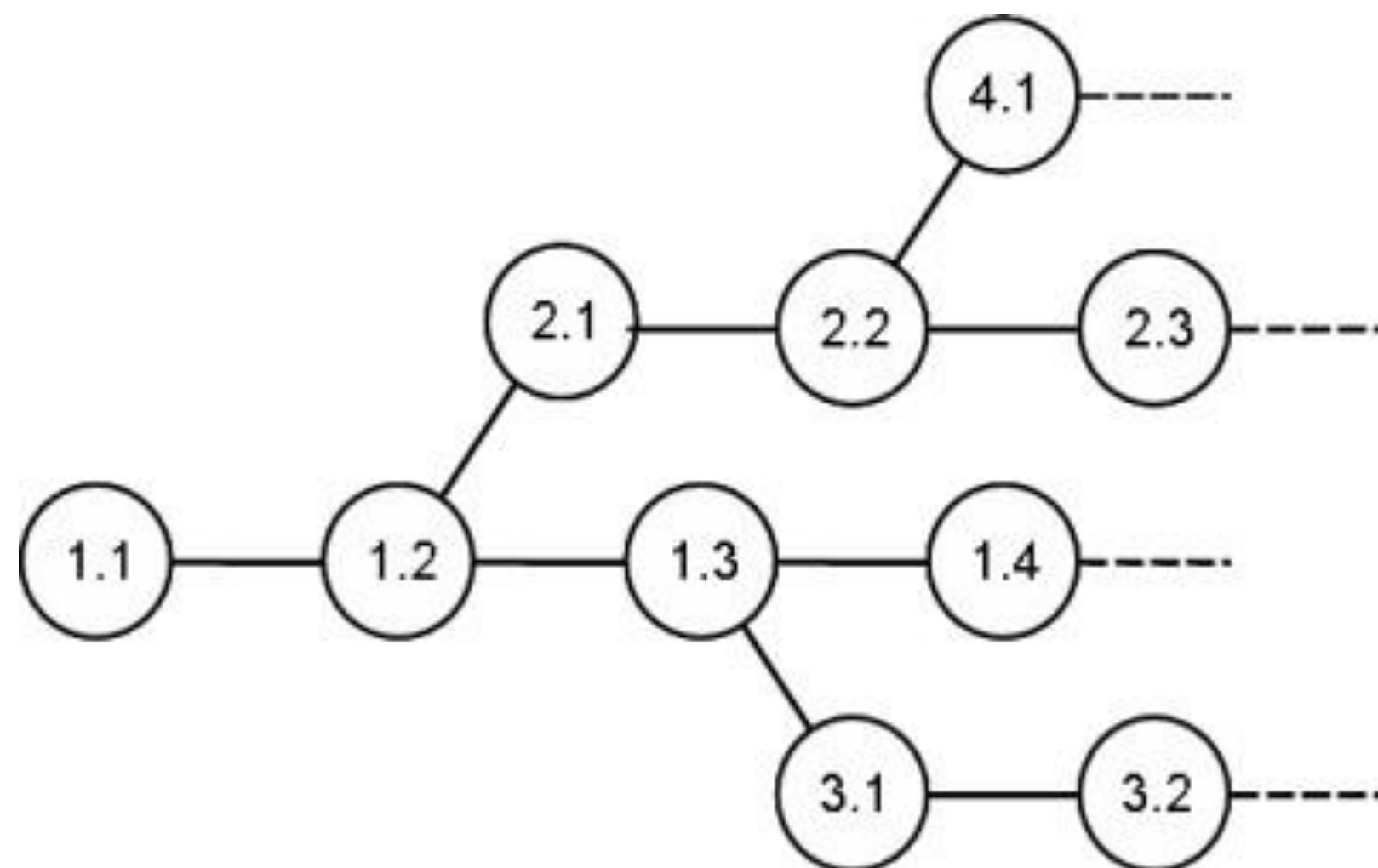


*Evolución lineal de un ítem de configuración*

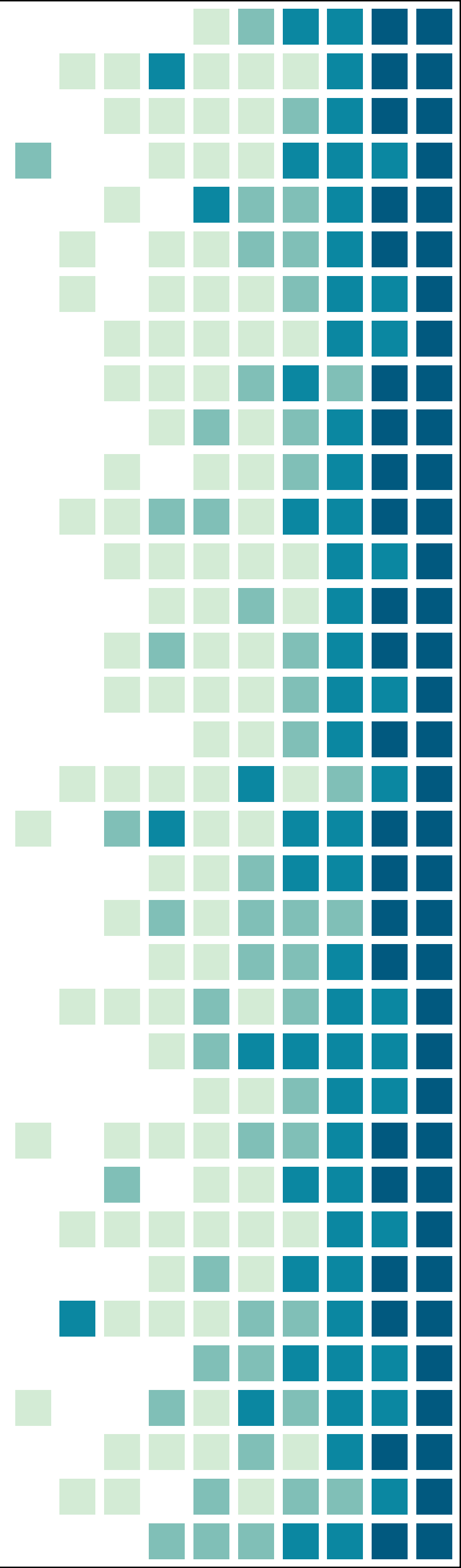


# Variante

- ❖ Una variante es una versión de un ítem de configuración (o de la configuración) que evoluciona por separado.
- ❖ Las variantes representan configuraciones alternativas.
- ❖ Un producto de software puede adoptar distintas formas (configuraciones) dependiendo del lugar donde se instale.
- ❖ Por ejemplo, dependiendo de la plataforma (máquina + S.O.) que la soporta, o de las funciones opcionales que haya de realizar o no.



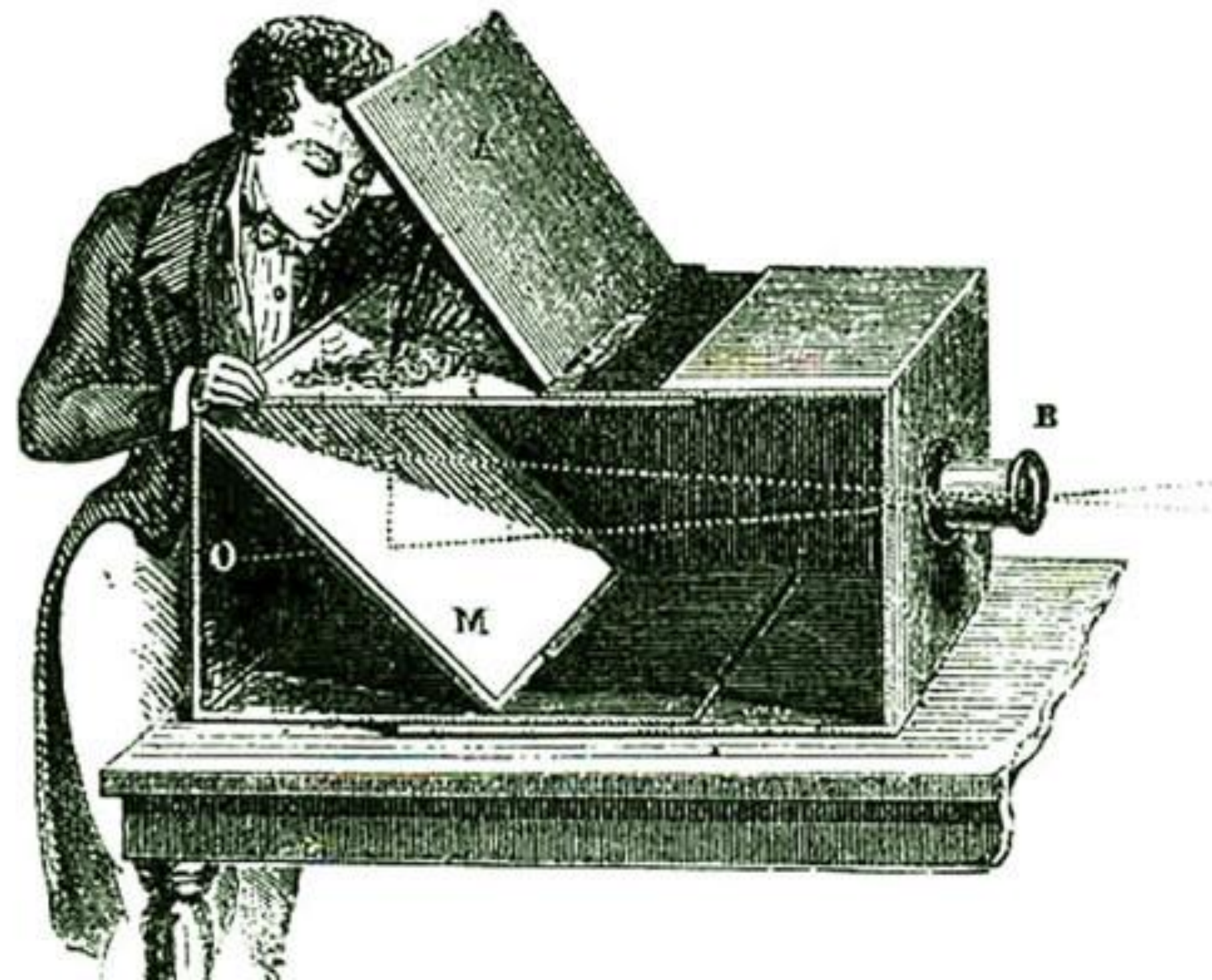
*Variante de un ítem de configuración*





# La Configuración del Software

Un conjunto de ítems de configuración con su correspondiente versión en un momento determinado





# ¿Qué es un Repositorio?



- ❖ Un repositorio de información conteniendo los ítems de configuración (ICs)
- ❖ Mantiene la historia de cada IC con sus atributos y relaciones.
- ❖ Usado para hacer evaluaciones de impacto de los cambios propuestos.
- ❖ Pueden ser una o varias bases de datos

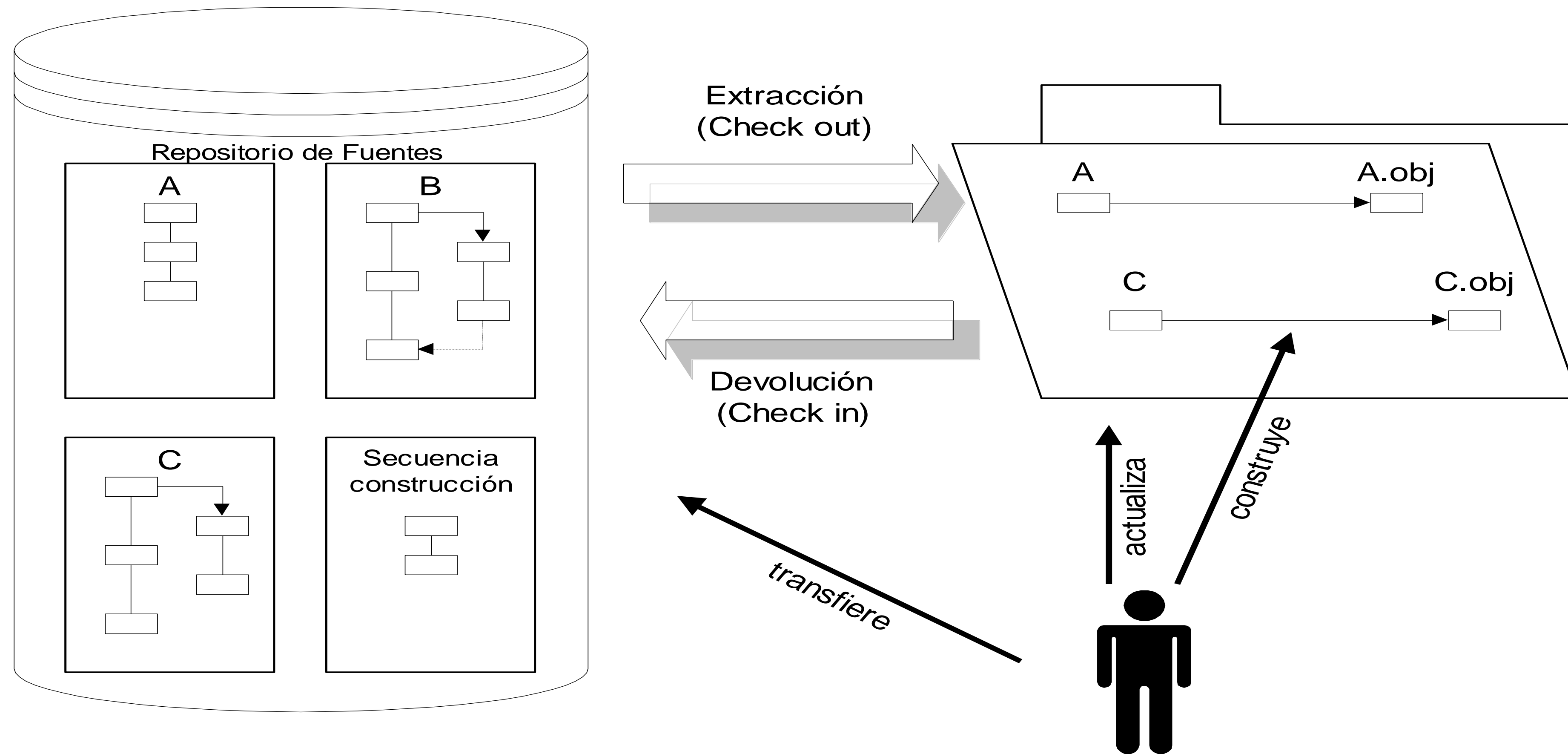




## 23

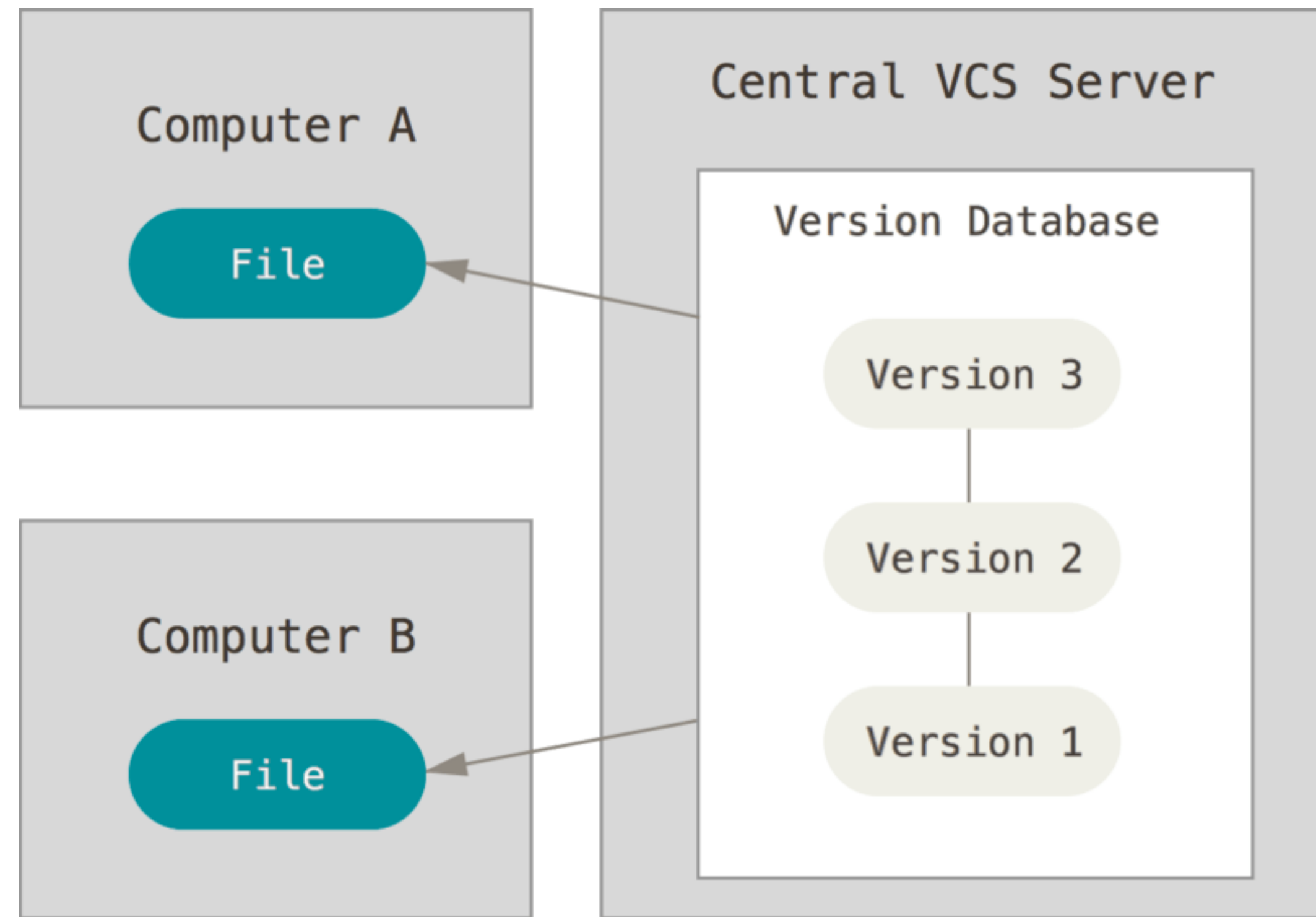


# Funcionamiento del Repositorio



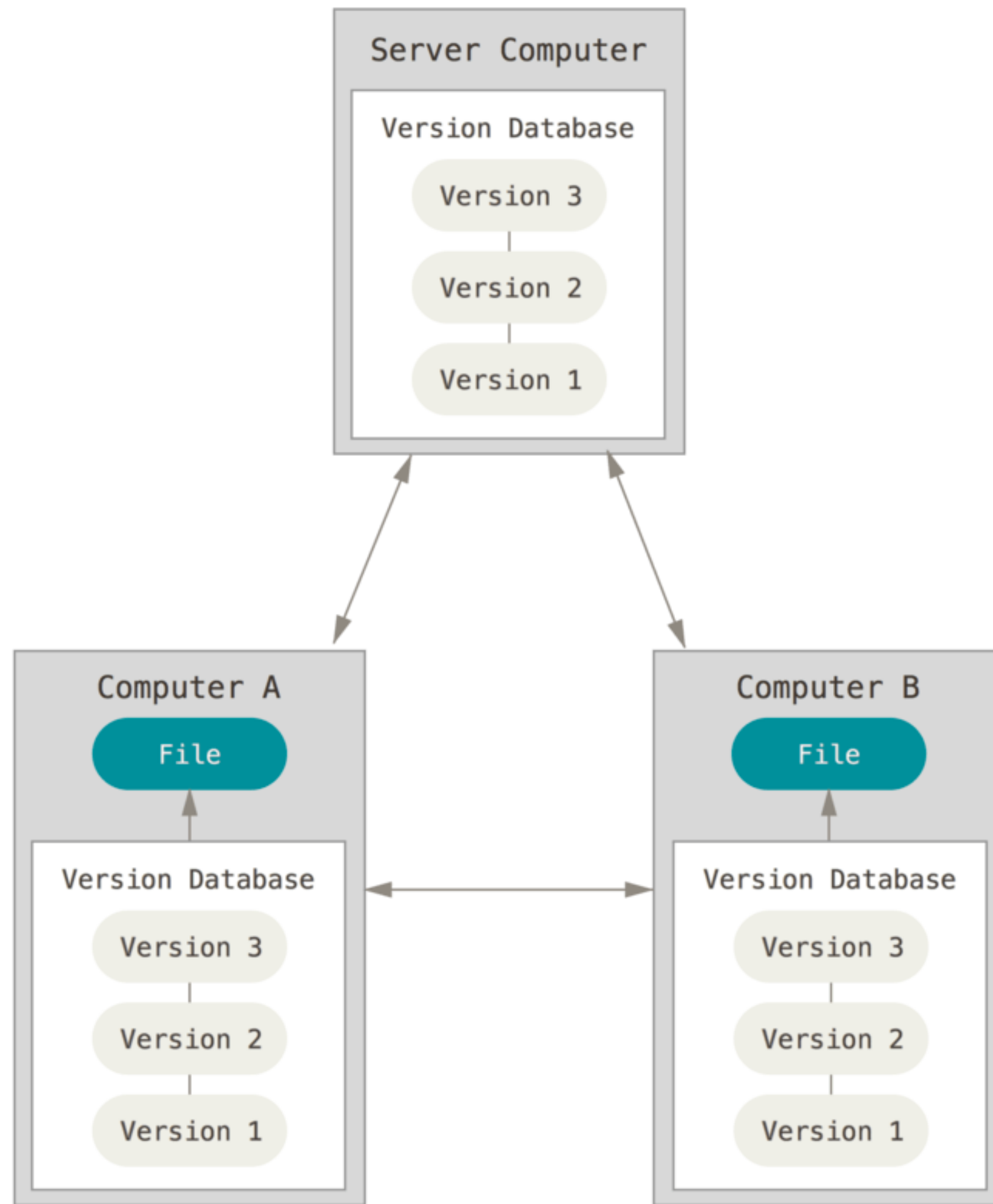


# Repositorios Centralizados



- ❖ Un servidor contiene todos los archivos con sus versiones.
- ❖ Los administradores tiene mayor control sobre el repositorio.
- ❖ Falla el servidor y "estamos al horno".

# Repositorios Descentralizados



- ❖ Cada cliente tiene una copia **exactamente** igual del repositorio completo.
- ❖ Si un servidor falla sólo es cuestión de "copiar y pegar".
- ❖ Posibilita otros workflows no disponibles en el modelo centralizado.



# Identificación de la Línea Base

- ❖ Se utilizan etiquetas para “marcar” las baseline
- ❖ No confundir con la versión del Producto





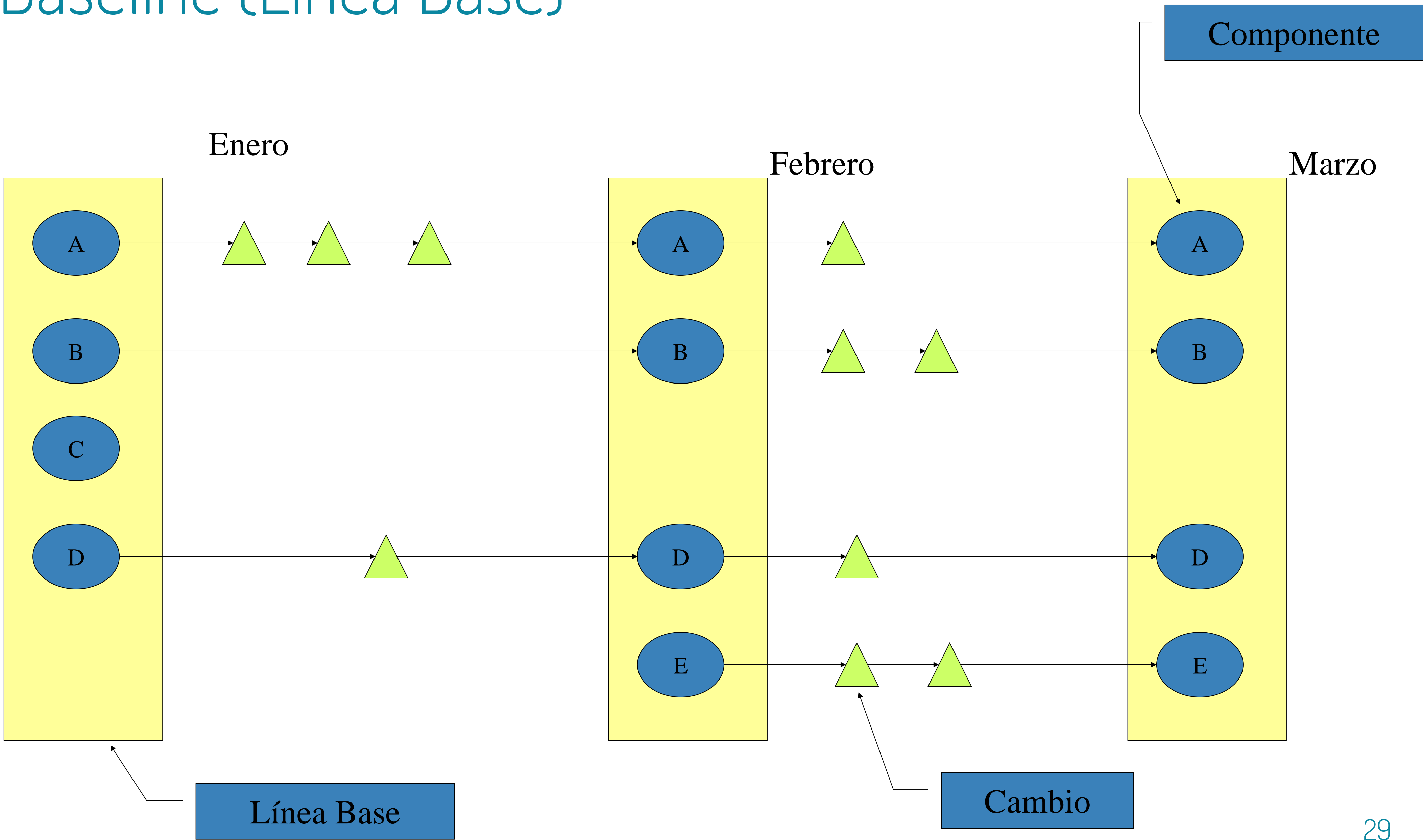
# Líneas Base



- ❖ Una configuración que ha sido revisada formalmente y sobre la que se ha llegado a un acuerdo
- ❖ Sirve como base para desarrollos posteriores y puede cambiarse sólo a través de un procedimiento formal de control de cambios
- ❖ Permiten ir atrás en el tiempo y reproducir el entorno de desarrollo en un momento dado del proyecto



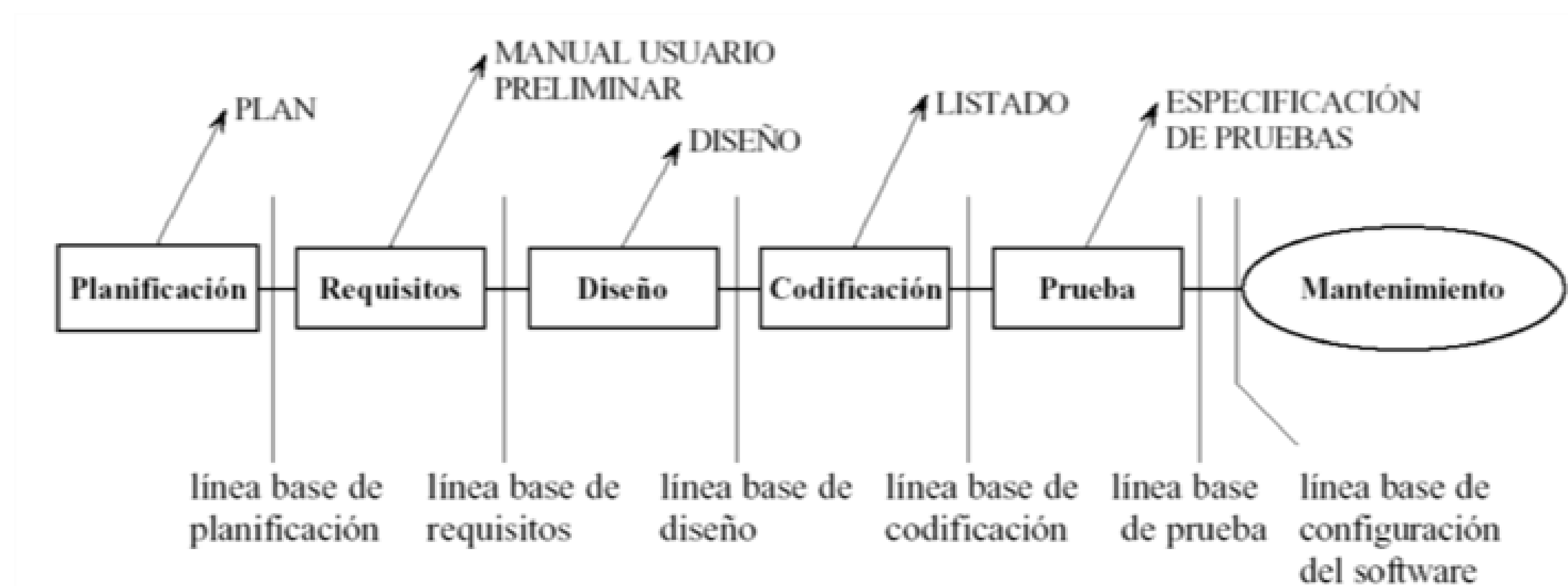
# Baseline (Línea Base)



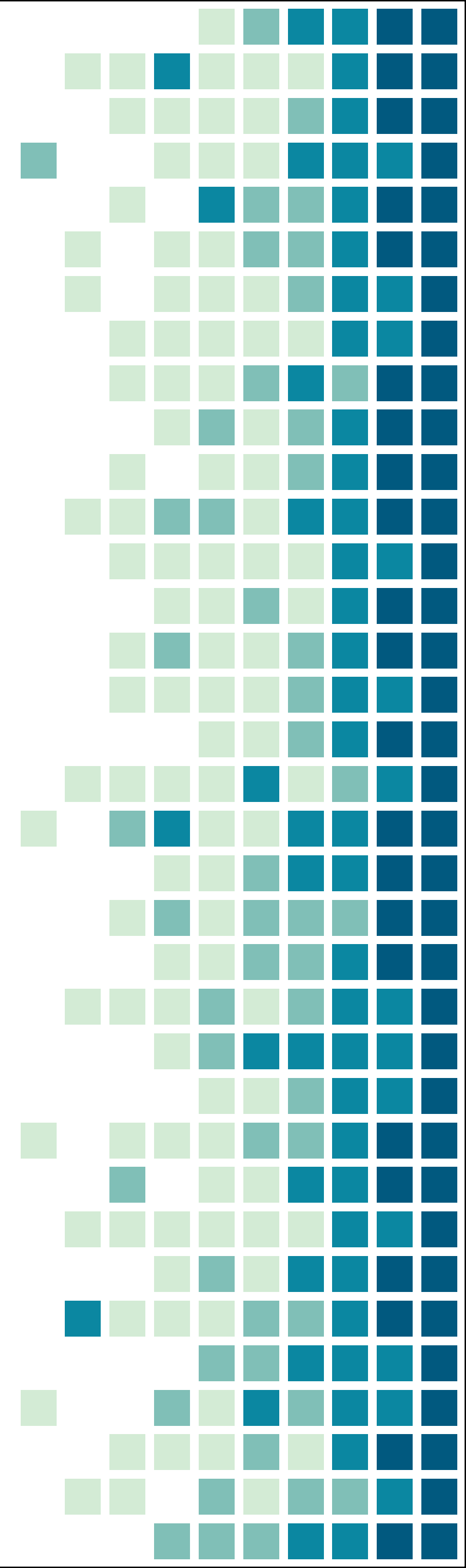
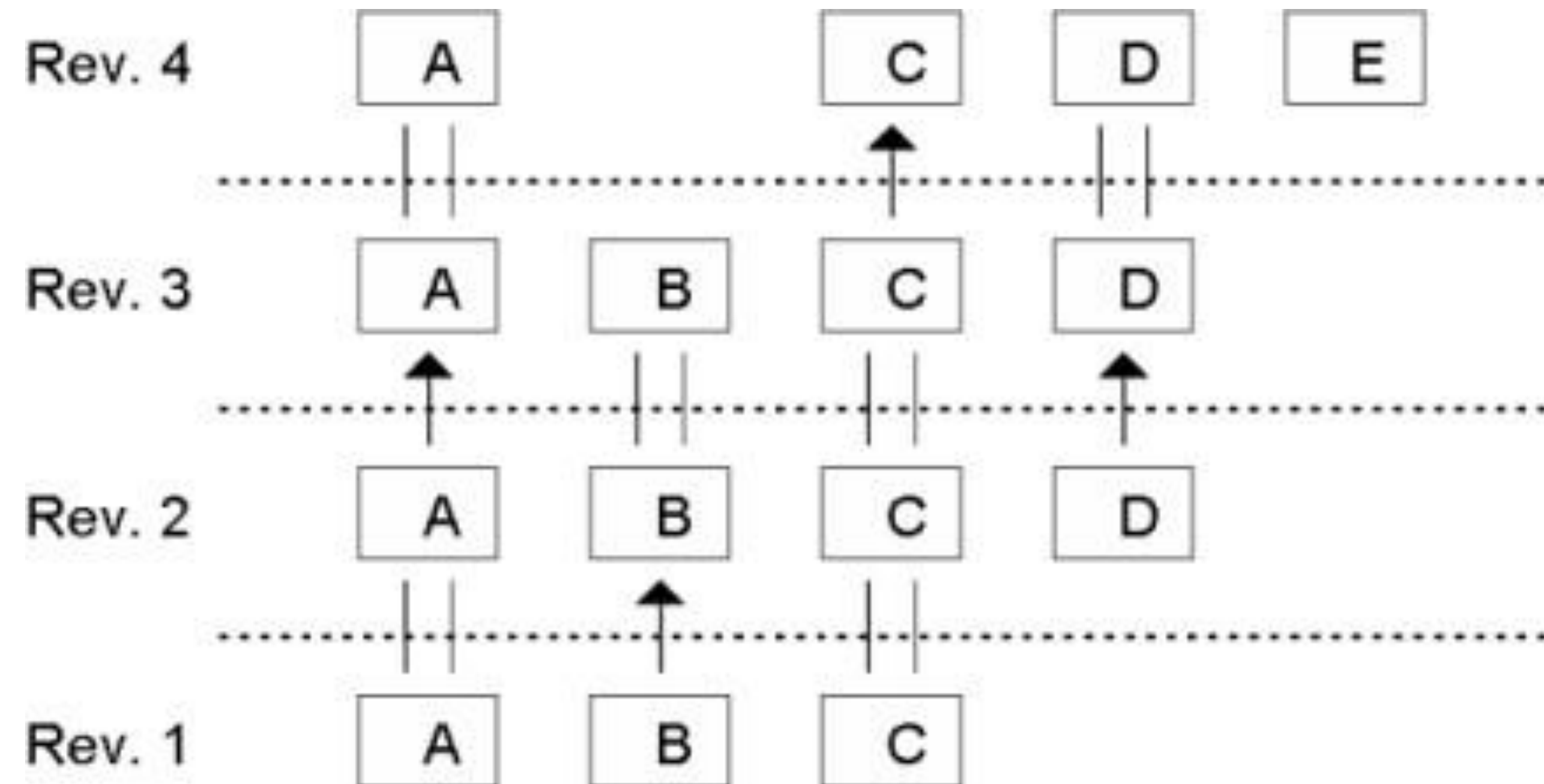
# Representación de Líneas Base

Pueden ser:

- De especificación (Requerimientos, Diseño)
- De productos que han pasado por un control de calidad definido previamente



# Evolución de una configuración





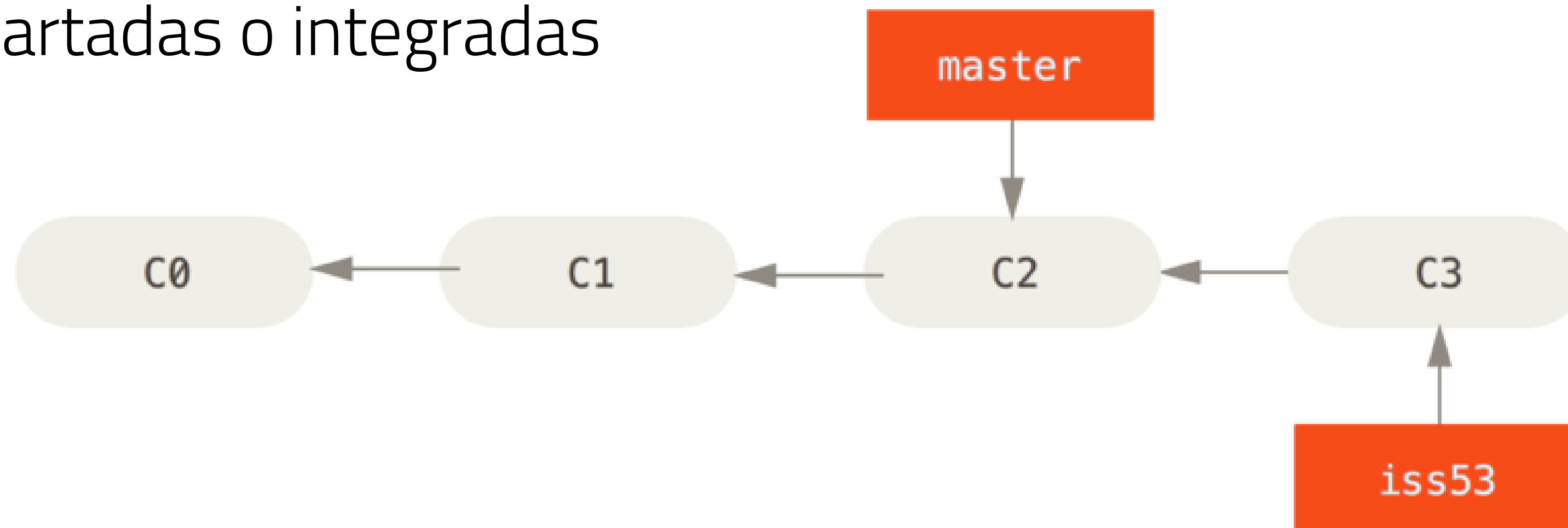


Ramas



# Creación de ramas

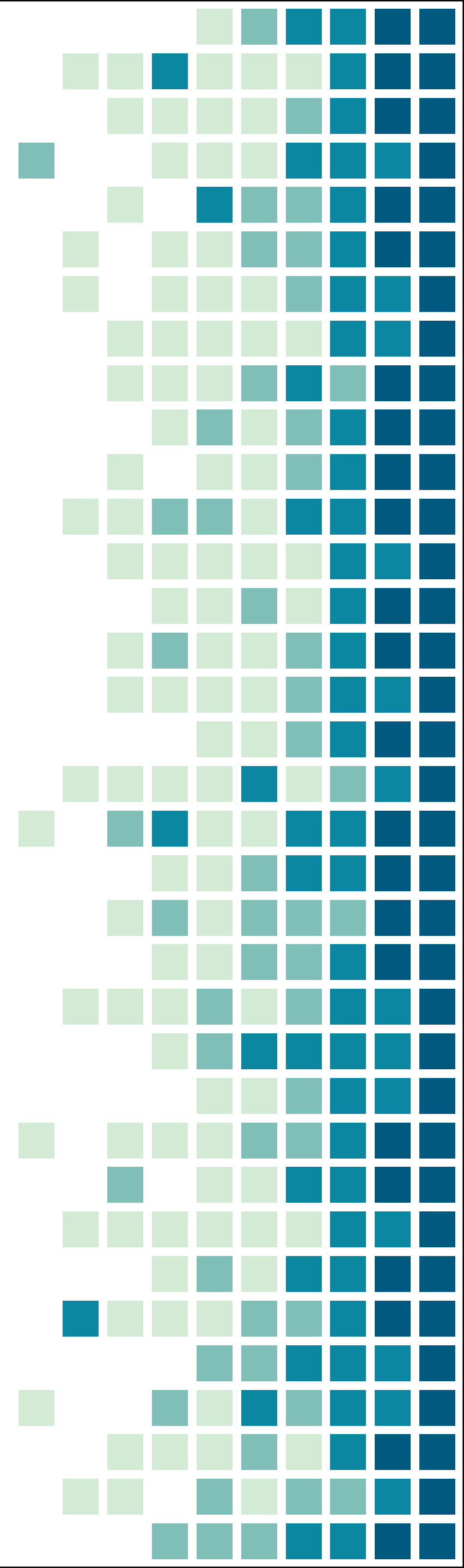
- ❖ Existe una rama principal (trunk, master)
- ❖ Sirven para bifurcar el desarrollo
- ❖ Pueden tener razones de creación con semántica
- ❖ Permiten la experimentación
- ❖ Pueden ser descartadas o integradas




# Integración de ramas



- ❖ La operación se llama merge
- ❖ Lleva los cambios a la rama principal
- ❖ Pueden surgir conflictos (resolvemos con diff)
- ❖ Todas las ramas deberían eventualmente integrarse a la principal o ser descartadas



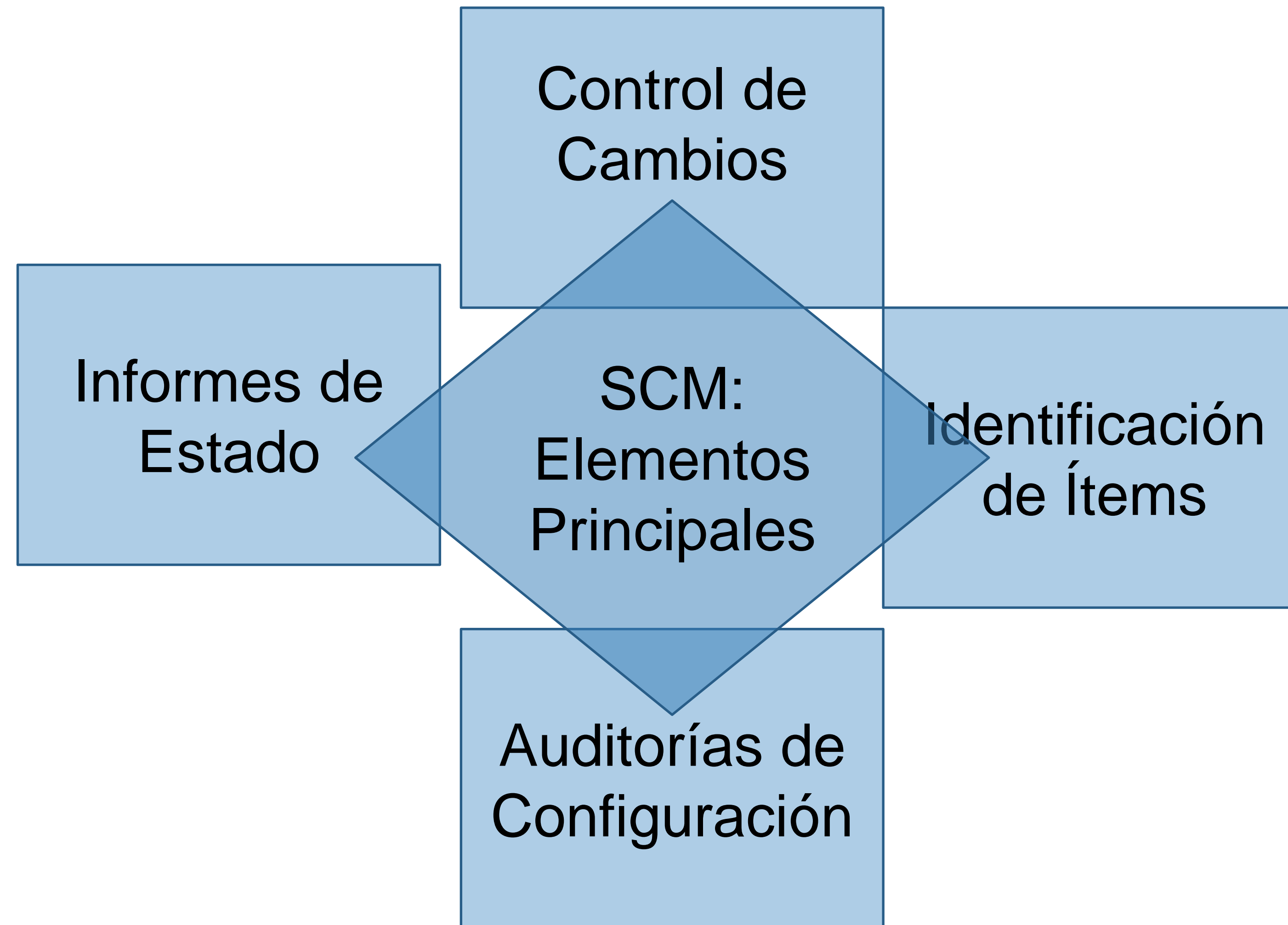
# Definición de Gestión de Configuración de Software

Una disciplina que aplica dirección y monitoreo administrativo y técnico a: identificar y documentar las características funcionales y técnicas de los ítems de configuración, controlar los cambios de esas características, registrar y reportar los cambios y su estado de implementación y verificar correspondencia con los requerimientos 

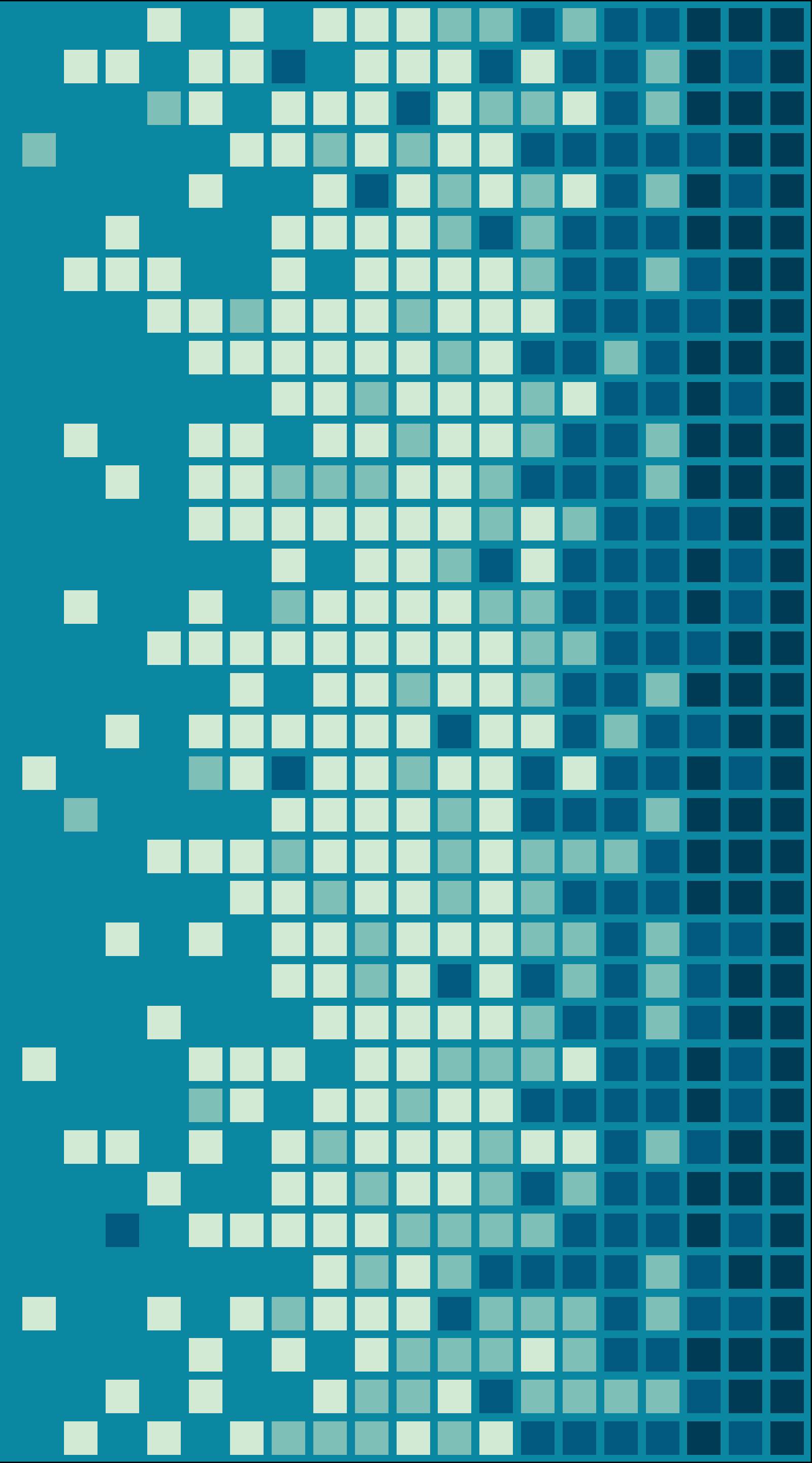
(ANSI/IEEE 828, 1990)



# Actividades Fundamentales de la Gestión de Configuración de Software

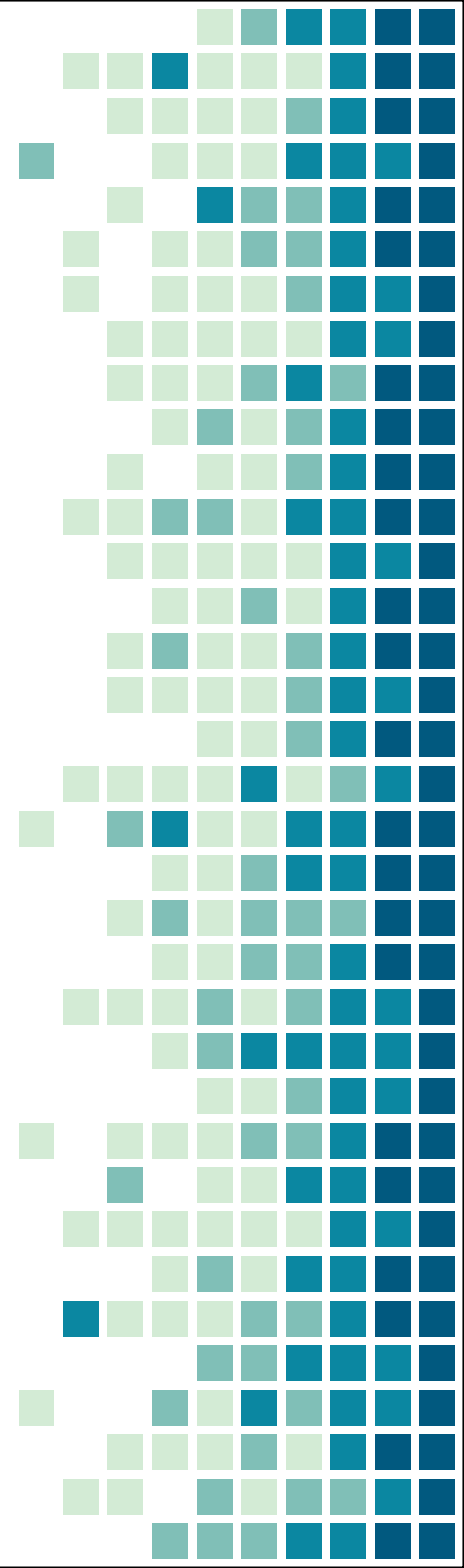


“ . *Identificación de ítems  
de configuración*



# Identificación de ítems de configuración

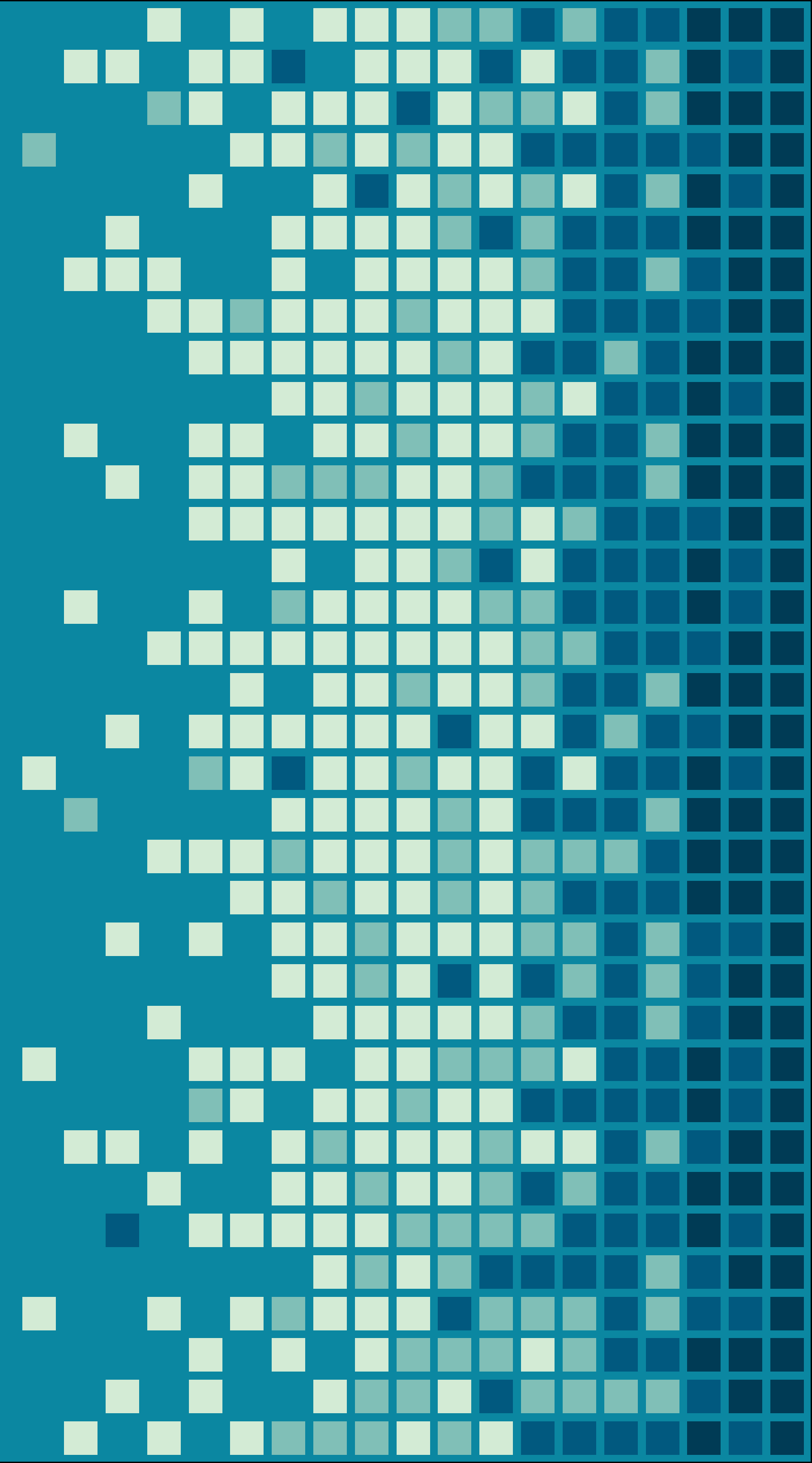
- ❖ Identificación unívoca de cada ítem de configuración
- ❖ Convenciones y reglas de nombrado
- ❖ Definición de la Estructura del Repositorio
- ❖ Ubicación dentro de la estructura del repositorio



# Ítems de Configuración para un proyecto de desarrollo de software

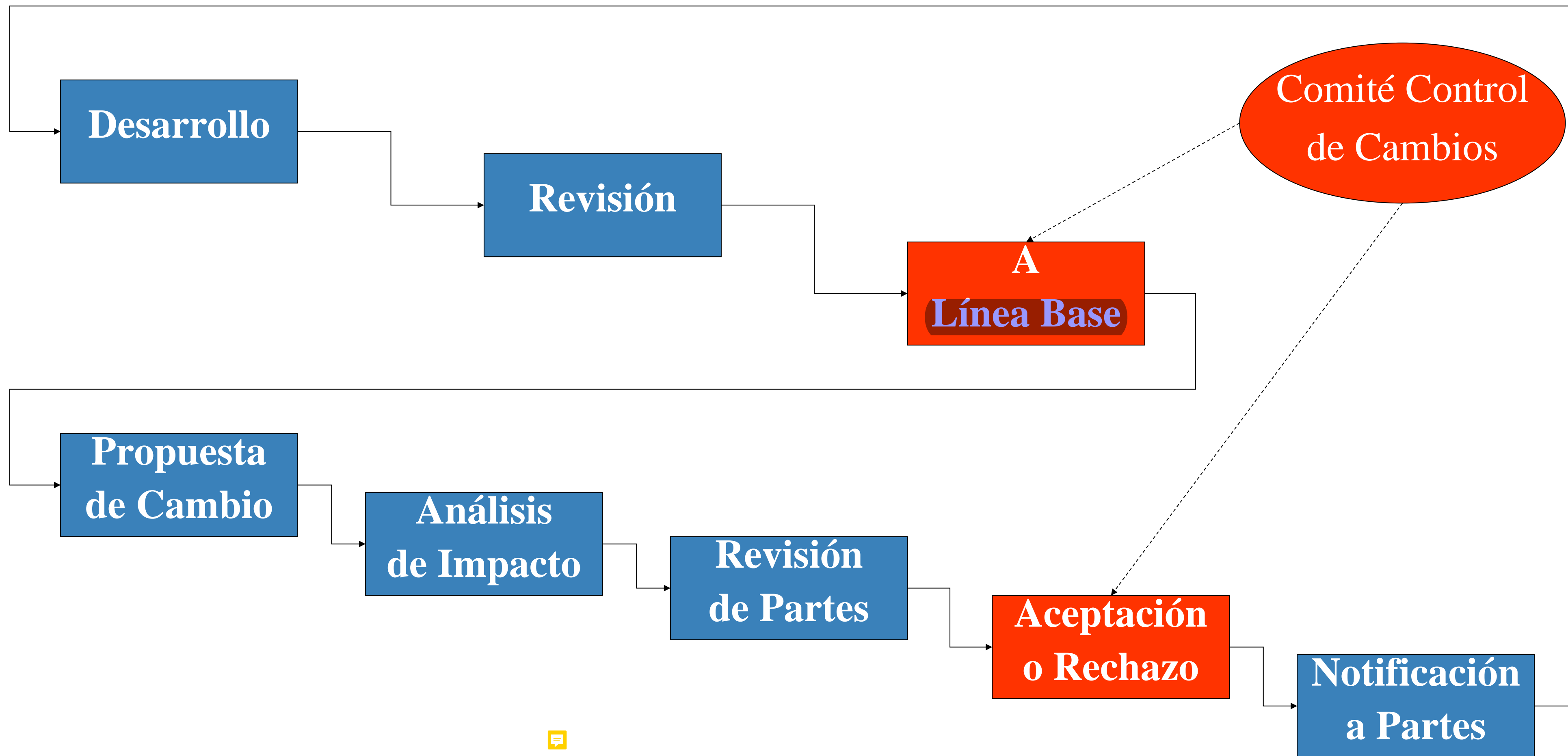


# “ Control de Cambios



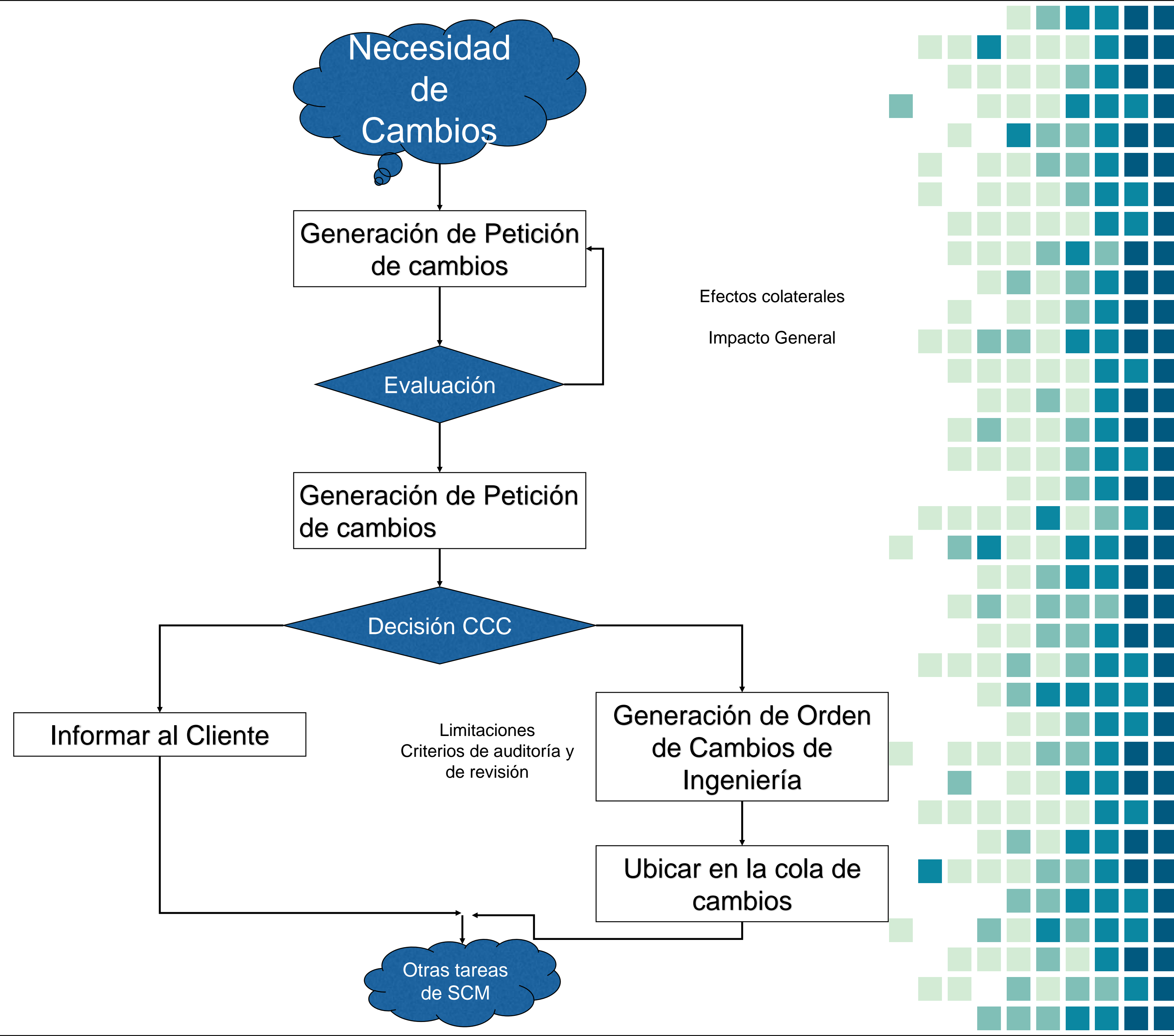


# Proceso de Control de Cambios

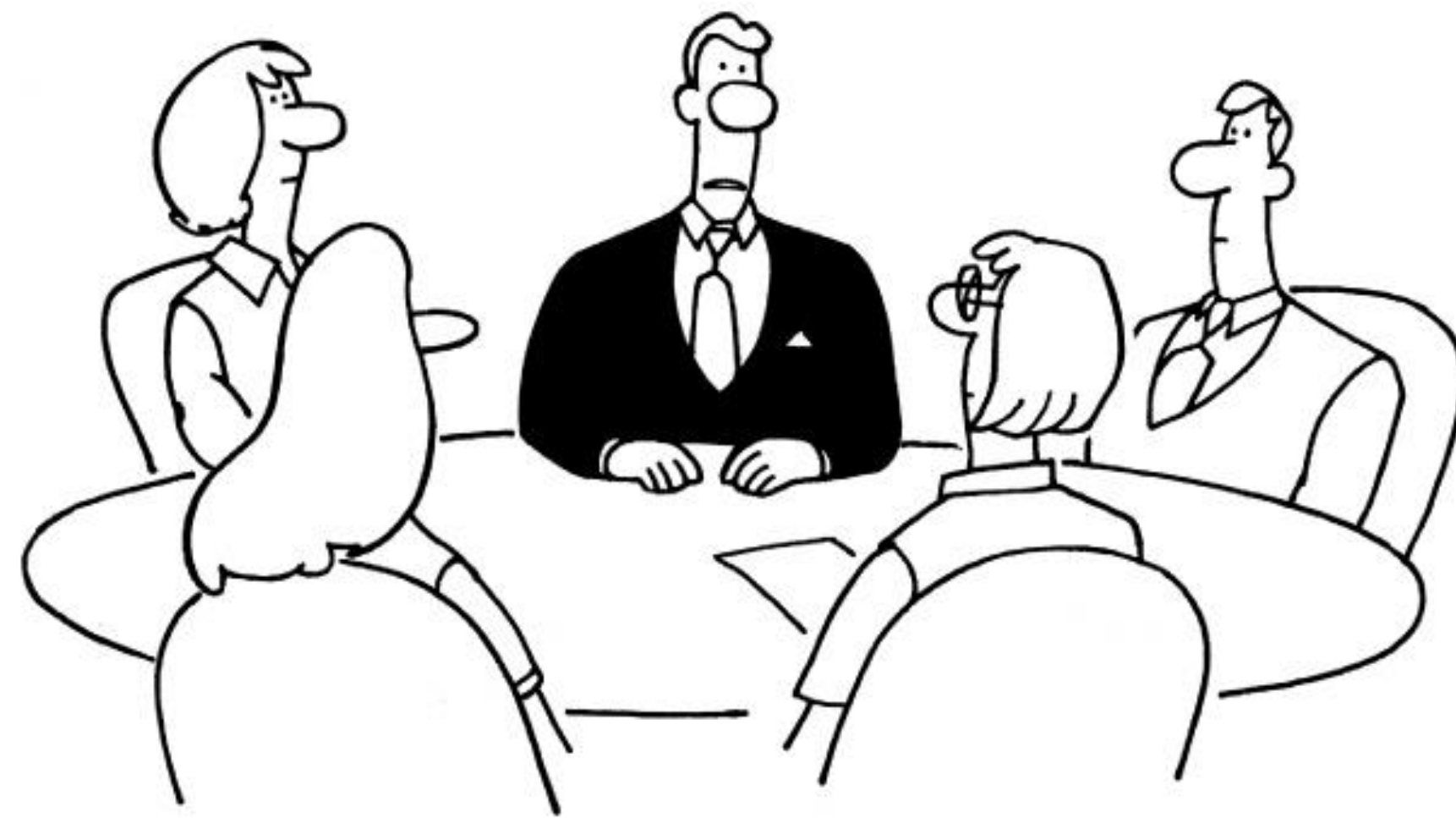


# Control de Cambios

- ❖ Tiene su origen en un Requerimiento de Cambio a uno o varios ítems de configuración que se encuentran en una **línea base**.
- ❖ Es un Procedimiento formal que involucra diferentes actores y una evaluación del **impacto** del cambio



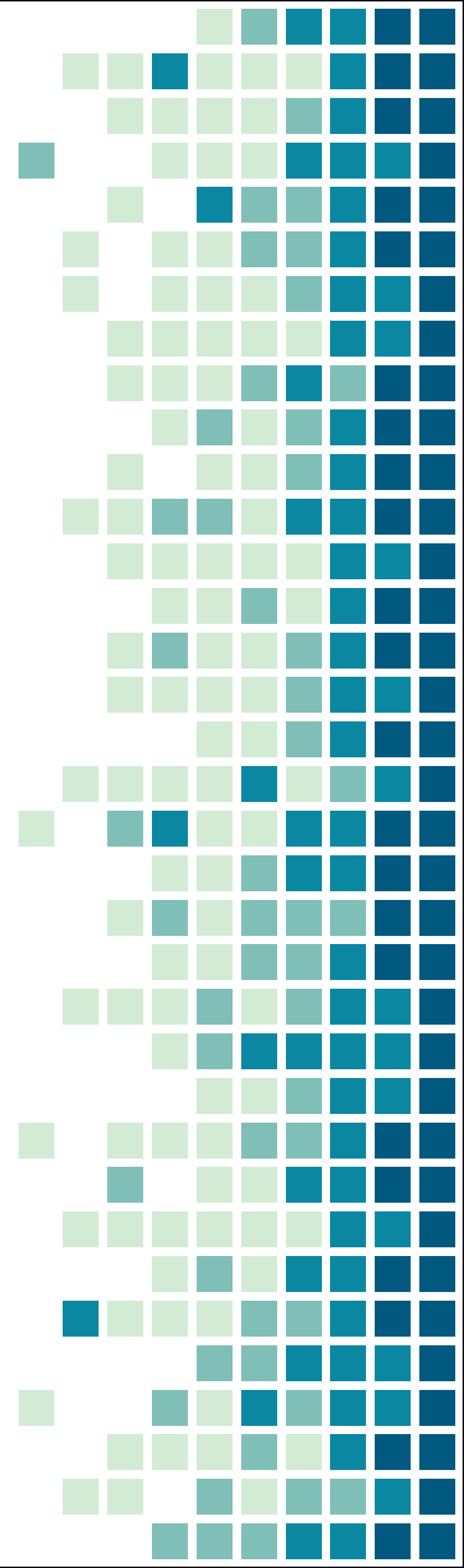
# El Comité de Control de Cambios



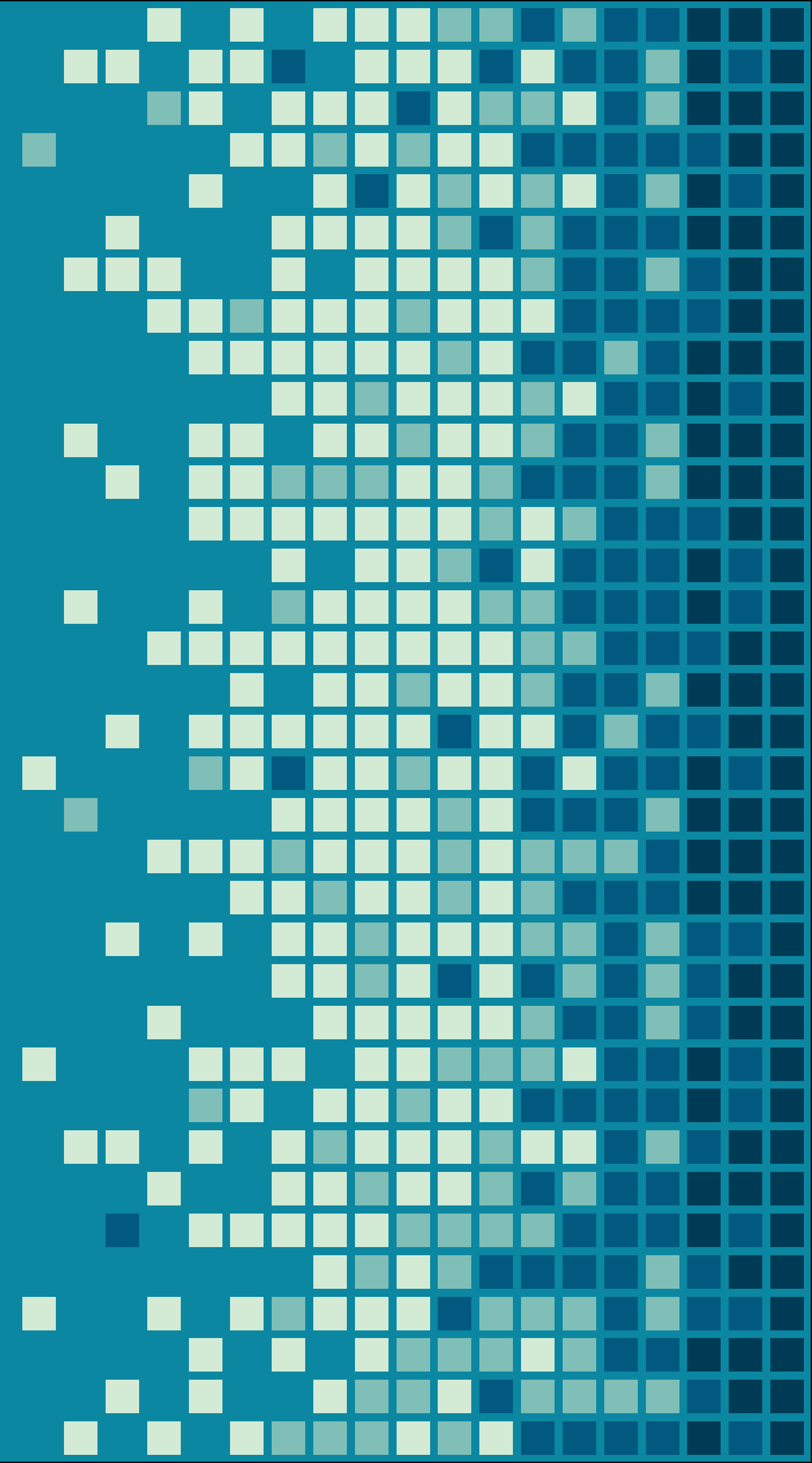
"Whew! That was close!  
We almost decided something!"

Está formado por representantes de todas las áreas involucradas en el desarrollo:

- ❖ Análisis, Diseño
- ❖ Implementación
- ❖ Testing
- ❖ Otros interesados



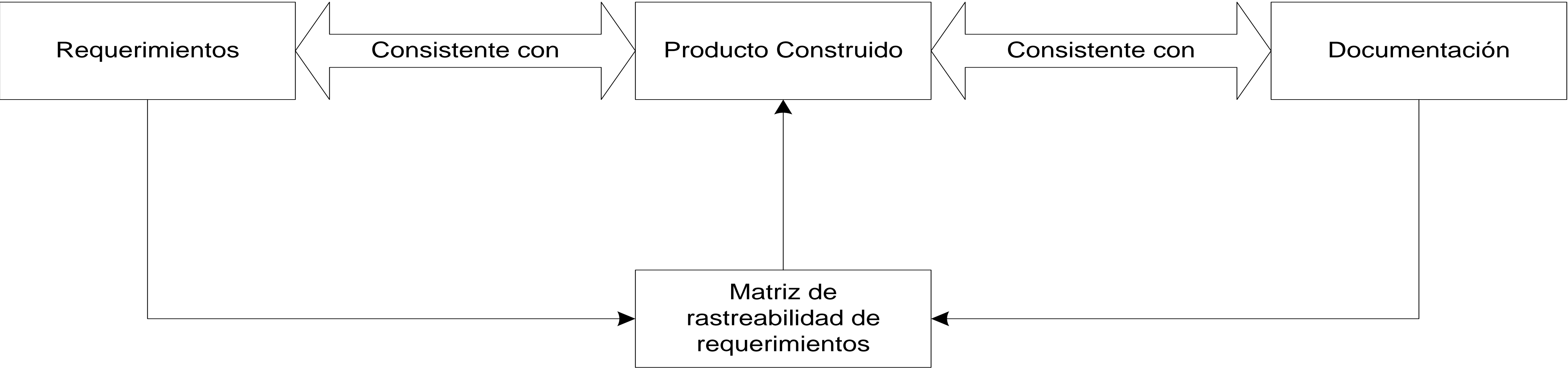
# “ . Auditorías de Configuración de Software



# Auditoría de Gestión de Configuración

Auditoría Funcional de Configuración

Auditoría Física de Configuración



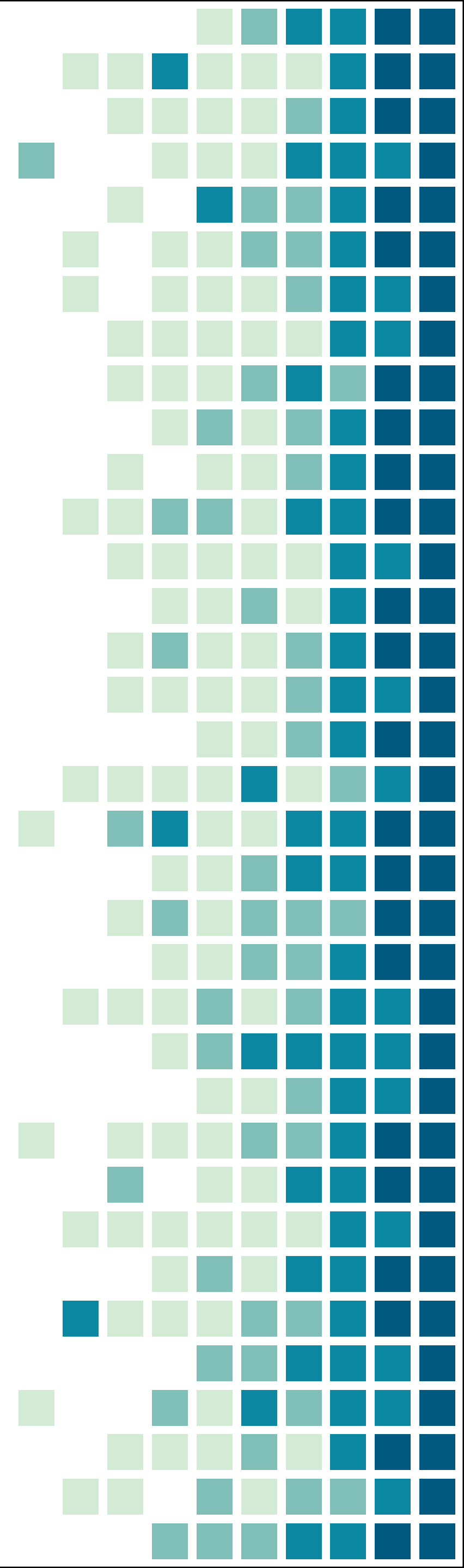
# Auditorías de Configuración

- ❖ **Auditoría física de configuración (PCA)**

Asegura que lo que está indicado para cada ICS en la línea base o en la actualización se ha alcanzado realmente. 🗨️

- ❖ **Auditoría funcional de configuración (FCA)**

Evaluación independiente de los productos de software, controlando que la funcionalidad y performance reales de cada ítem de configuración sean consistentes con la especificación de requerimientos. 🗨️

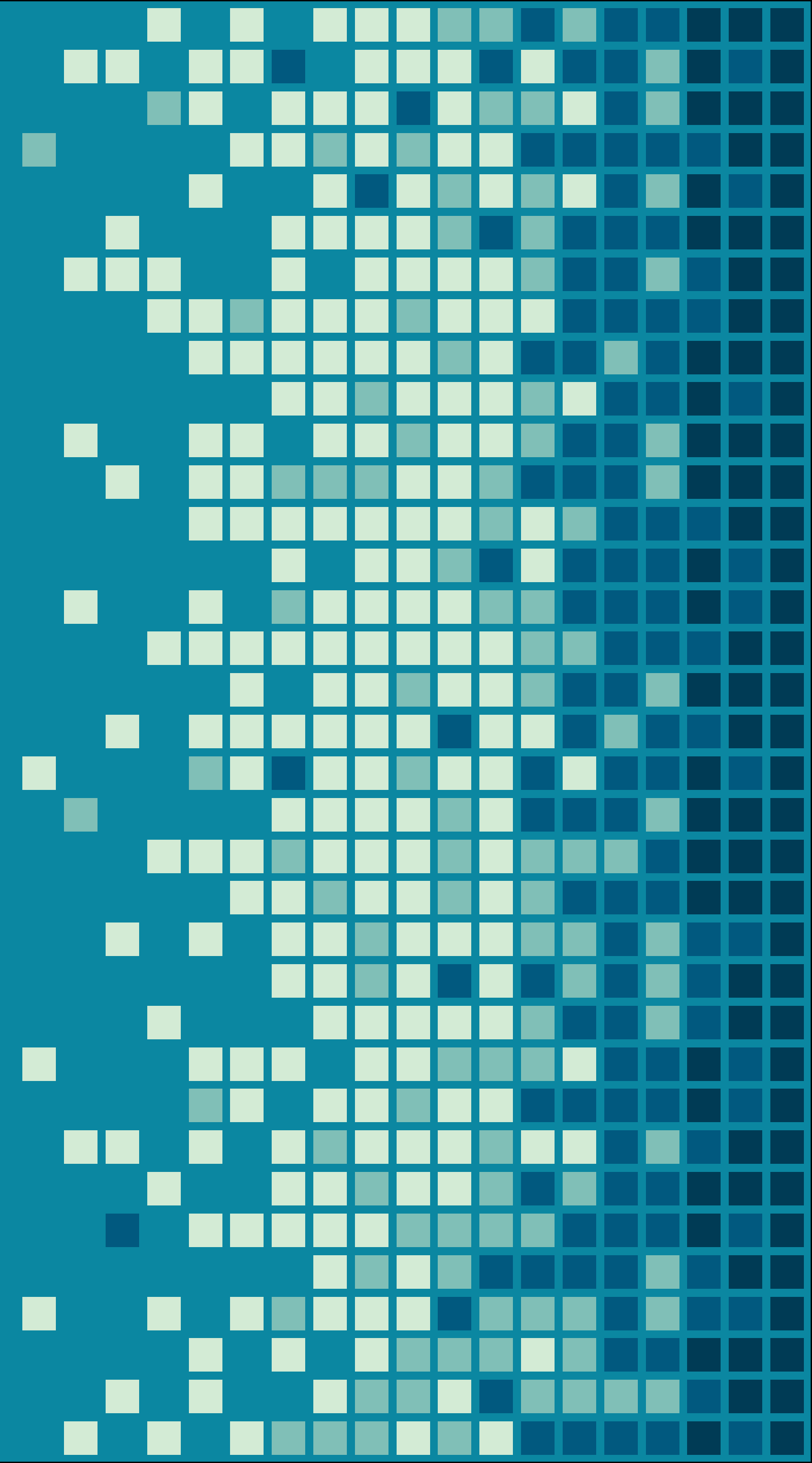


# Auditoría de Gestión de Configuración y V&V

Sirve a dos procesos básicos: la validación y la verificación

- ❖ **Validación:** el problema es resuelto de manera apropiada que el usuario obtenga el producto correcto.
- ❖ **Verificación:** asegura que un producto cumple con los objetivos preestablecidos, definidos en la documentación de líneas base (línea base). Todas las funciones son llevadas a cabo con éxito y los test cases tengan status "ok" o bien consten como "problemas reportados" en la nota de release.

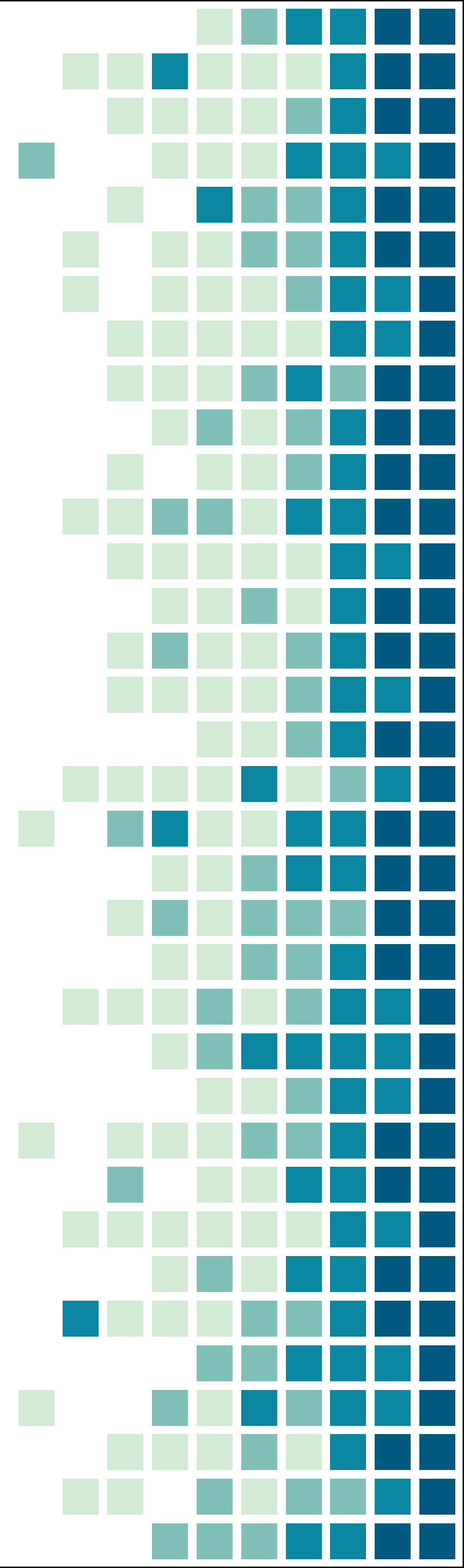
“ *Informes de Estado*





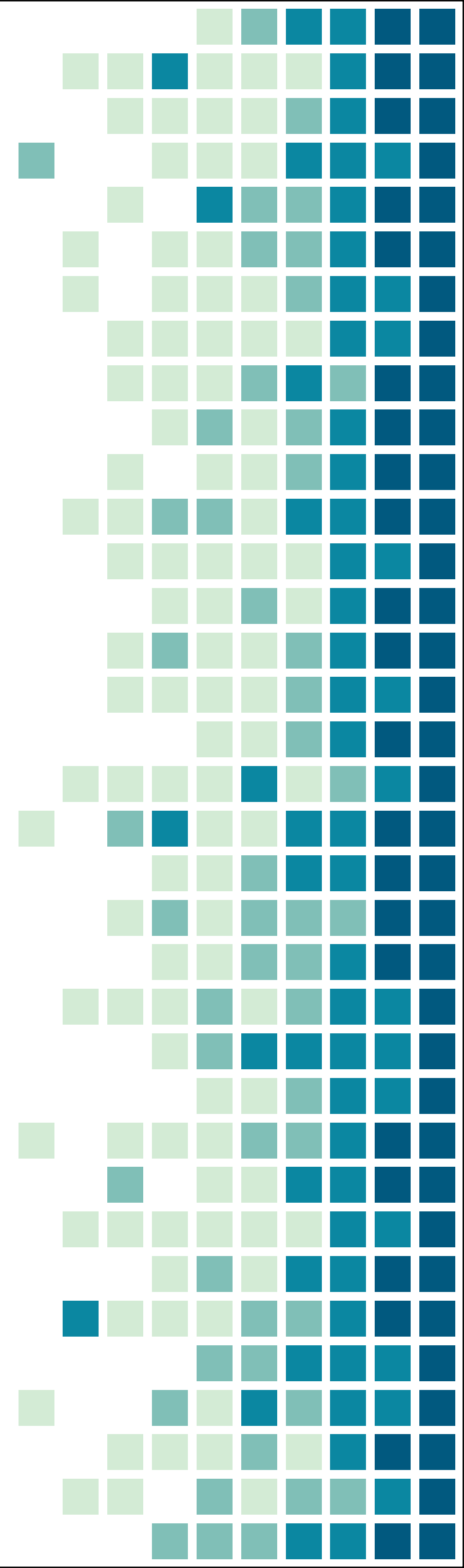
# Registro e Informe de Estado

- ❖ Se ocupa de mantener los registros de la evolución del sistema.
- ❖ Maneja mucha información y salidas por lo que se suele implementar dentro de procesos automáticos.
- ❖ Incluye reportes de rastreabilidad de todos los cambios realizados a las líneas base durante el ciclo de vida.



# Algunas preguntas que podría responder

- ❖ ¿Cuál es el estado del ítem?
- ❖ ¿Un requerimiento de cambio ha sido aprobado o rechazado por el CCB?
- ❖ ¿Qué versión de ítem implementa un requerimiento de cambio aprobado (saber cuál es el componente que contiene la mejora)?
- ❖ ¿Cuál es la diferencia entre una versión y otra dada?

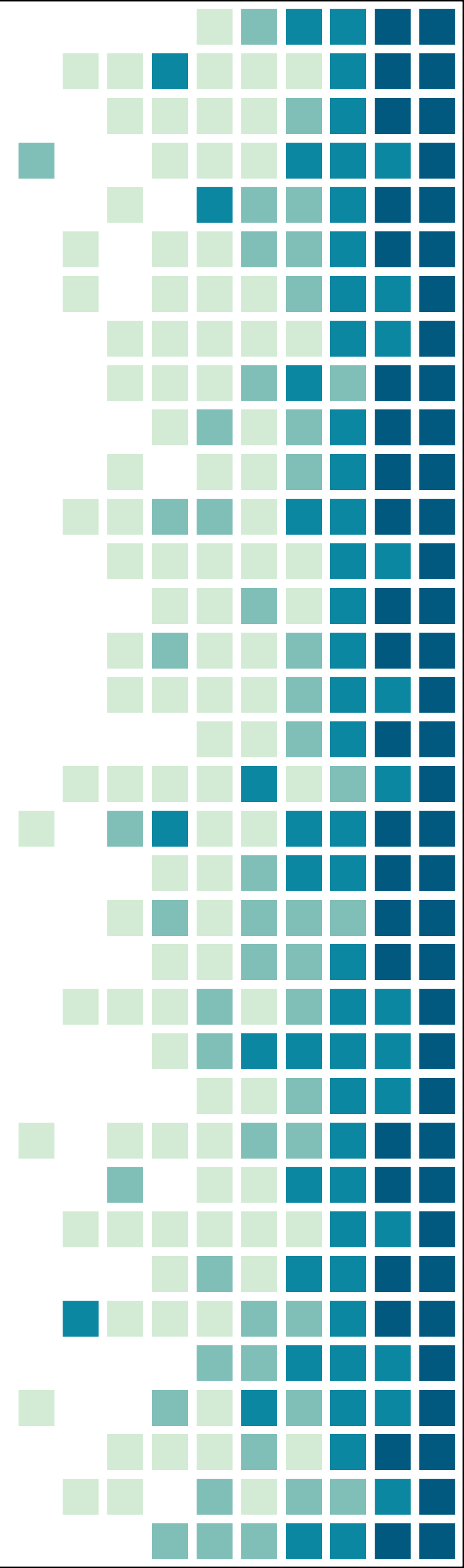




# Plan de Gestión de Configuración

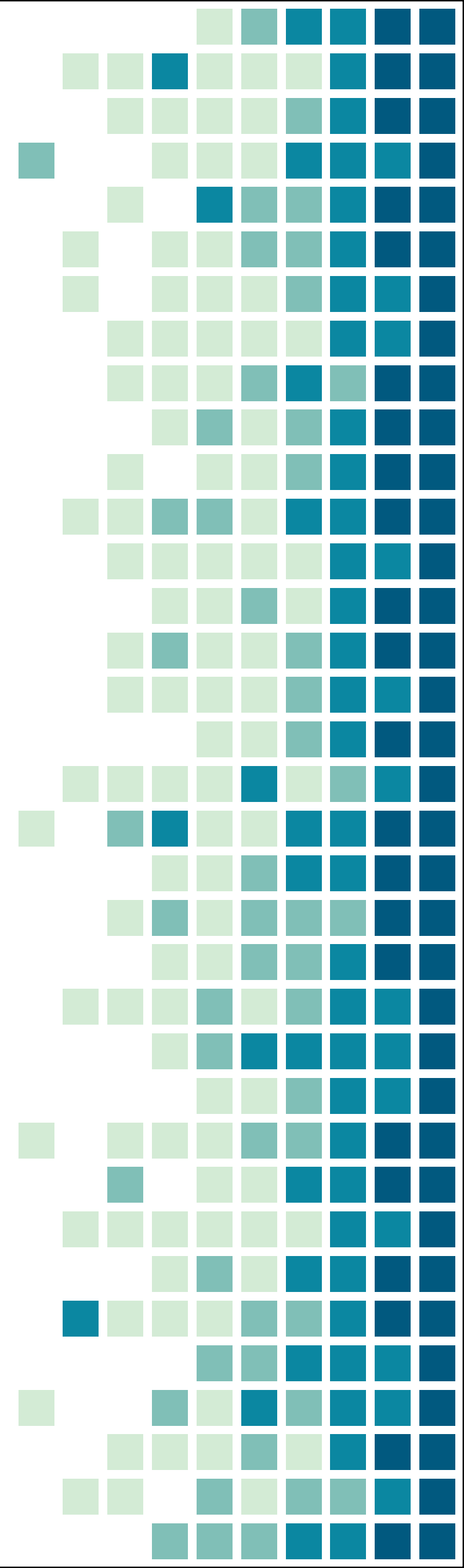
¡También se planifica! ¿Qué debería incluir el plan?

- ❖ Reglas de nombrado de los CI
- ❖ Herramientas a utilizar para SCM
- ❖ Roles e integrantes del Comité
- ❖ Procedimiento formal de cambios
- ❖ Plantillas de formularios
- ❖ Procesos de Auditoría



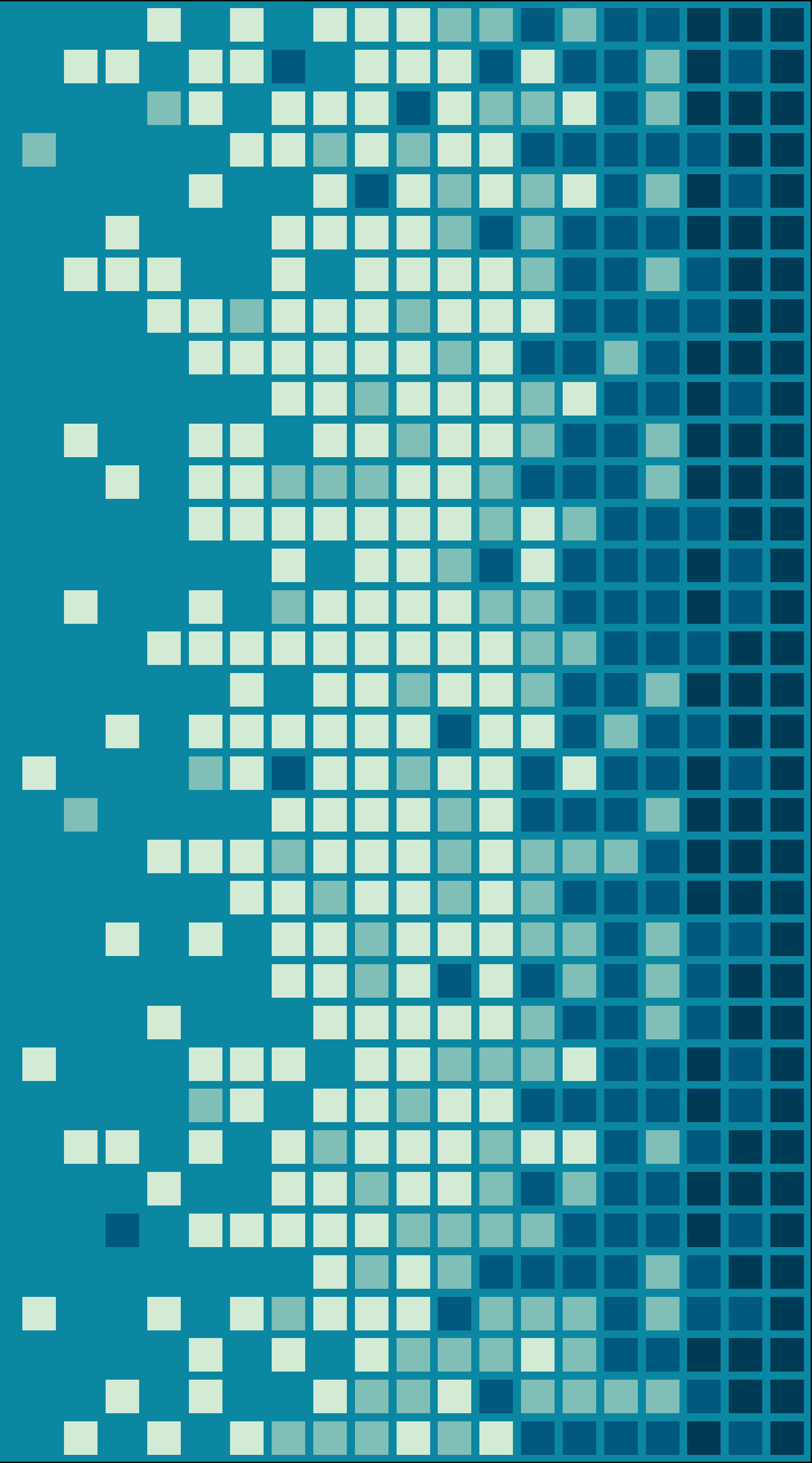
# Plan de Gestión de Configuración

- ❖ Definir estructura del repo
- ❖ Identificar ICs
  - Nombre (nomenclatura)
  - Que es
  - Ubicacion del Archivo
- ❖ Glosario
- ❖ Criterio de la LB
- ❖





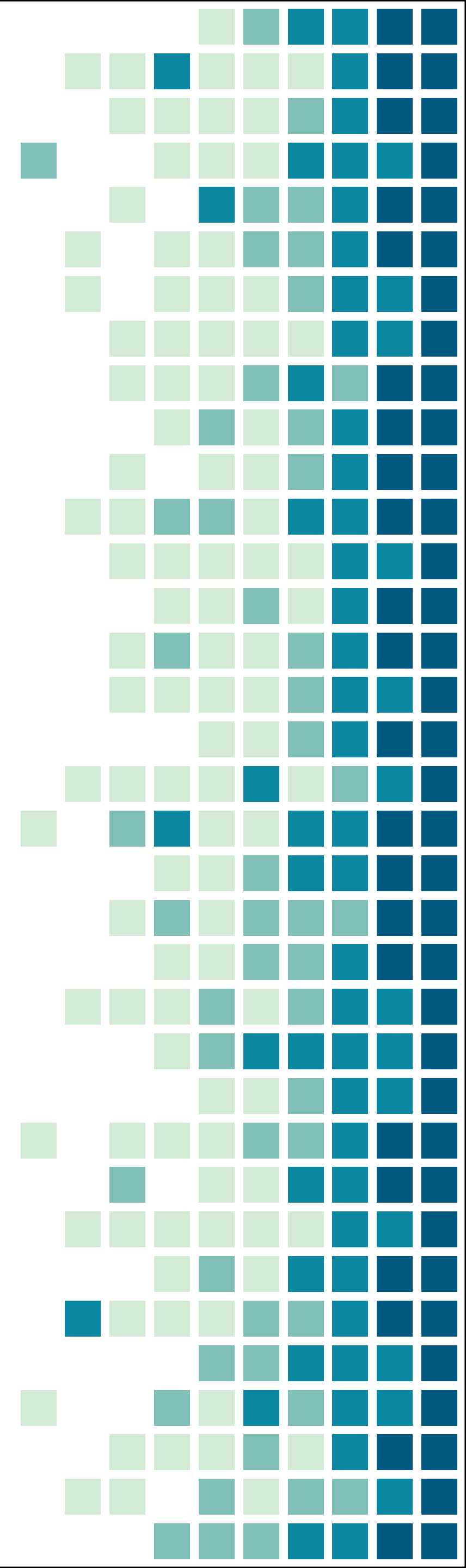
“ . *Gestión de Configuración  
de Software en  
ambientes Ágiles*



Recuerdan...

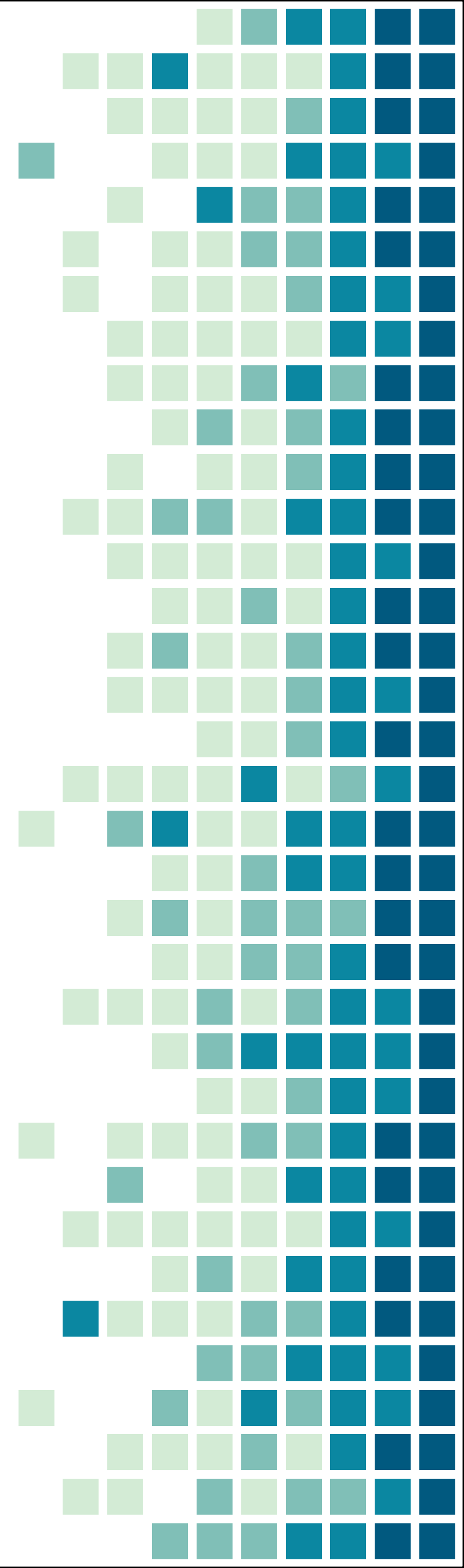
Manifiesto

Ágil



# SCM en Agile

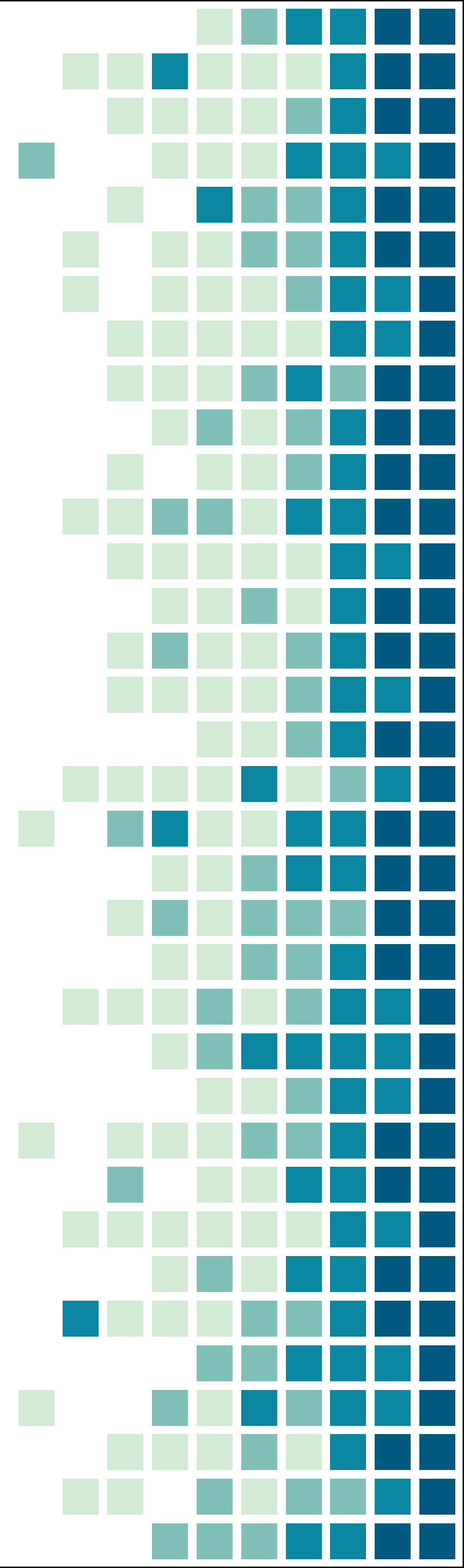
- ❖ Sirve a los practicantes (equipo de desarrollo) y no viceversa.
- ❖ Hace seguimiento y coordina el desarrollo en lugar de controlar a los desarrolladores.
- ❖ Responde a los cambios en lugar de tratar de evitarlos.
- ❖ Esforzarse por ser transparente y "sin fricción", automatizando tanto como sea posible.
- ❖ Coordinación y automatización frecuente y rápida.
- ❖ Eliminar el desperdicio - no agregar nada más que valor.
- ❖ Documentación Lean y Trazabilidad.
- ❖ Feedback continuo y visible sobre calidad, estabilidad e integridad





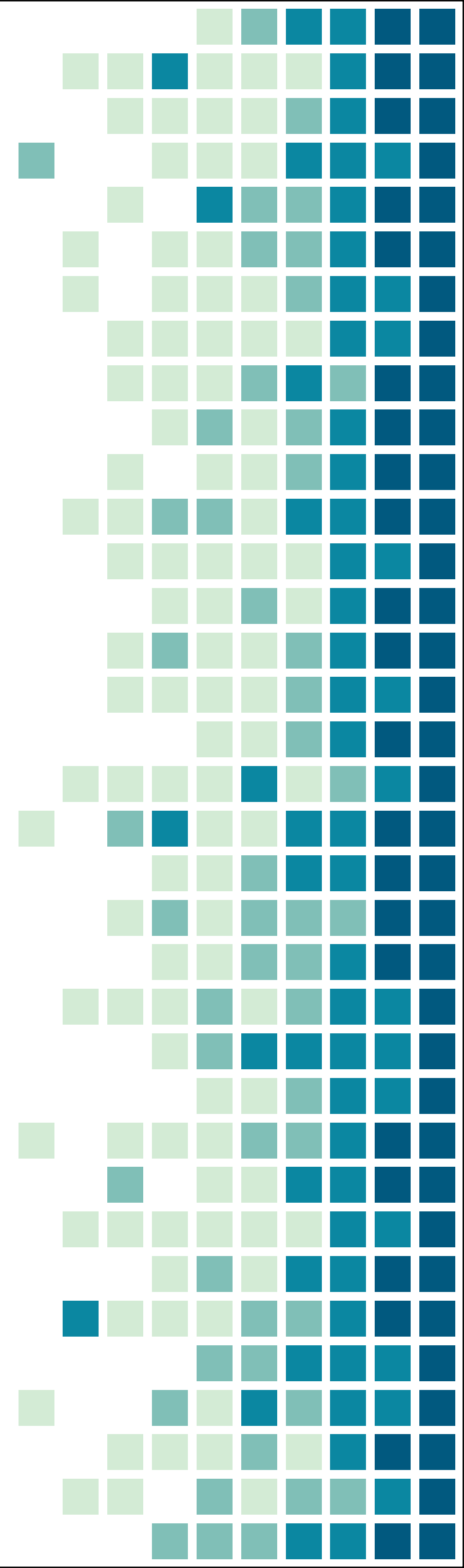
# SCM en Agile, algunos tips....

- ❖ Es responsabilidad de todo el equipo.
- ❖ Automatizar lo más posible.
- ❖ Educar al equipo.
- ❖ Tareas de SCM embebidas en las demás tareas requeridas para alcanzar el objetivo del Sprint.



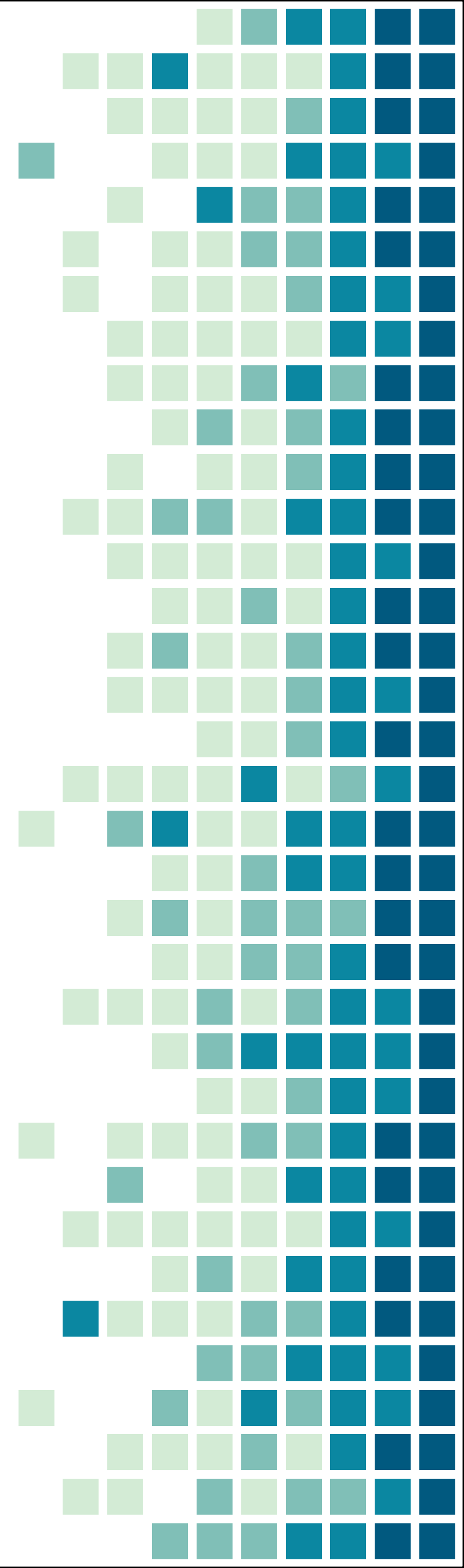
# SCM en Agile, para debatir....

- ❖ ¿Qué pasa con el Comité de Control de Cambios?
- ❖ ¿Qué ítems de configuración podemos tener?
- ❖ ¿Qué pasa con las auditorías?
- ❖ ¿Qué pasa con los reportes de estado?



# Referencias

- Bersoff, E.H., "Elements of Software Configuration Management",
- IEEE Transactions on Software Engineering, vol 10, nro. 1, enero 1984, pp 79-87
- Little Book of Configuration Management – <http://www.spmn.com>
- SCM & the Agile Manifesto - <http://www.scmpatterns.com/agilescm/>
- Harness.io - <https://harness.io/blog/continuous-verification/blue-green-canary-deployment-strategies/>



# Integración, Entrega y Despliegue Continuos

